Reliable Navigation of Mobile Sensors in Wireless Sensor Networks without Localization Service

Qingjun Xiao, Bin Xiao, Jiaqing Luo and Guobin Liu
Department of Computing
The Hong Kong Polytechnic University
Hunghom, Kowloon, Hong Kong
E-mail: {csqjxiao, csbxiao, csjluo, csgliu}@comp.polyu.edu.hk

***Abstract*—This paper deals with the problem of guiding mobile sensors (or robots) to a phenomenon across a region covered by static sensors. We present a distributed, reliable and energy-efficient algorithm to construct a smoothed moving trajectory for a mobile robot. The reliable trajectory is realized by first constructing among static sensors a *distributed hop count based artificial potential field* (DH-APF) with only one local minimum near the phenomenon, and then navigating the robot to that minimum by an attractive force following the reversed gradient of the constructed field. Besides the attractive force towards the phenomenon, our algorithm adopts an additional repulsive force to push the robot away from obstacles, exploiting the fast sensing devices carried by the robot. Compared with previous navigation algorithms that guide the robot along a planned path, our algorithm can (1) tolerate the potential deviation from a planned path, since the DH-APF covers the entire deployment region; (2) mitigate the trajectory oscillation problem; (3) avoid the potential collision with obstacles; (4) save the precious energy of static sensors by configuring a large moving step size, which is not possible for algorithms neglecting the issue of navigation reliability. Our theoretical analysis of the above features considers practical sensor network issues including radio irregularity, packet loss and radio conflict. We implement the proposed algorithm over TinyOS and test its performance on the simulation platform with a high fidelity provided by TOSSIM and Tython. Simulation results verify the reliability and energy efficiency of the proposed mobile sensor navigation algorithm.**

***Index Terms*—Hybrid Sensor Networks, Mobile Sensors Navigation, Location Free, Angle of Arrival, Navigation Reliability.**

I. Introduction

Hybrid sensor networks comprising of mobile sensors and cheap static sensors open new frontiers in a variety of military and civilian applications. In hybrid sensor networks, a large quantity of networked static sensors monitor the environment, while a few mobile sensors provide the actuation with locomotion modules equipped. Typical usages of these mobile nodes include energy recharge for static sensors, collection of a large volume of data in Delay Tolerant Networks, counterattack against intruders. As a summary, the emergence of hybrid sensor networks converts the static sensor networks for environmental monitoring into a reactive or active system [1].

One fundamental problem for the hybrid sensor networks is how to guide a mobile sensor to a phenomenon across the region covered by static sensors while evading the obstacles [1]. This problem is different from the past robot path planning problem based on centrally stored map, since the mobile sensors (or robots) utilize distributed sensing ability of static sensors to detect dangers (or obstacles) and their distributed computations plus wireless connections to plan the moving trajectories collaboratively. These distributed sensing and path planning abilities can enhance the robots' ability to function in unfamiliar environments, where the hybrid sensor networks has two advantages: easy-to-deploy (e.g., by a flying vehicles) and more accurate sensing ability. For example, within a forest, their distributed sensing abilities can penetrate through the thick vegetation to detect wild fire early, and in a battle field, they can not be easily deceived by enemies' disguise, if compared with the remote sensing based on satellites.

For the problem of mobile sensor navigation assisted by static sensors, one constraint is that an effective localization service for static sensors is not always available, especially in complex concave regions [2], [3]. Moreover, localization services may require additional ranging hardware that increases WSNs deployment cost, and running localization services may elongate the network deployment time, consumes the precious energy of static sensors. Considering these facts, we have this location-free assumption, which may invalidate some previous mobile sensor navigation schemes. For example, the Berkeley solution [4] of the pursuer-evader game depends on the static sensors with location knowledge to track the evader and guide the pursuer to the reported position by an installed map, which however may be unavailable in unfamiliar environments.

Several recent works [1], [5] can remove these location and map assumptions by running a location-free routing protocol within the network and by navigating the robots following the routing path to the sensor detecting the phenomenon. To make the routing path more tangible for robots, a navigation band is explicitly built by [5] along the path and the robot knows its relative position to the band by wireless communications. However, there are two inadequacies for these previous works. First, the robots are only guided by networks and the readings from sensors carried by the robots are forgotten, which can provide fast response to unexpected obstacles. Second, the guiding from networks depends on wireless communications, which are by nature unreliable (e.g., radio irregularity, lossy channels and radio conflicts). It's not yet clear how this communication unreliability affects the robots' navigation quality.

This paper mainly provides two contributions for this prob-

978-1-4244-3876-1/09/$25.00 ©2009 IEEE

lem of location-free robot navigation. **First**, through both analysis and simulation studies, we demonstrate that, for the navigation based on unreliable talks with the static sensor networks, the robots frequently make wrong decisions about their next step moving directions. The robots therefore may collide with obstacles, deviate from an planned path or move out of sensor deployment fields, which reduces the navigation qualities and consumes their energies. Moreover, the robots may suffer from path oscillation, caused by the narrowness of the navigation band. **Second**, we present a reliable robot navigation algorithm which can handle the unreliability in wireless communications, construct a smoothed robot moving trajectory and improve the reliability of robot navigation. Our algorithm is based on the *artificial potential field* (APF) [6] and adopts a two layer architecture [7] - a global path planning module over a local obstacle avoidance layer. The global planing is conducted by an event dissemination initiated by the static sensor detection an phenomenon. This dissemination constructs a *distributed hop count based APF* (DH-APF), which has only one local minimum near the phenomenon. Afterwards, the reversed gradients of this DH-APF can provide to the robots attractive forces leading them to the phenomenon. In practice, we propose to implement these attractive forces by an elegant integration of the robot navigation protocol and the tree based routing protocol with the pull rule. Additionally, the agile repulsive forces can be provided by the fast readings from the robots' sensors to steer the robots away from unexpected obstacles (called reactive local obstacle avoidance).

Our algorithm is different from the previous works [1], [8]–[10] that also deal with navigation qualities for the following reasons. (1) Their major quality metric is the exposure to danger problem, since they models the navigating robots either as malicious entities trying to avoid the tracking by static sensor networks or as cautious entities trying to keep the safest distance to obstacles. In contrast, our work regards that the duty of sensor networks is only to provide optimized navigation guidance to the goals and without the collision with obstacles, and that the obstacle avoidance should be the duty of robots themselves by their self-carried sensors. In this way, the static sensors can save the energy for disseminating the knowledge about obstacles and robots make their own decisions on avoiding obstacles (perhaps fast moving unexpected ones), since no matter how fast the wireless communications are in spreading out obstacles' knowledge, they can not compete with the speed of robots' self-carried sensors. (2) Their navigation quality evaluation is conducted assuming ideal sensor communication model. Although [1] also presents the imperfect radio channel problem, it does not present systematical analysis on how error-prone nature of sensor networks affects robots' navigation quality, which is one of the major focuses of this paper. (3) Previous works fails to consider the trajectory smoothness, which is one of the advantages of our algorithms, since we recommend to smooth the attractive forces of all upstream nodes rather than to follow the direction of only one upstream node on the planned path.

To verify the designed features of our navigation algorithm, we implement the behavior of static sensors over TinyOS [11] and the functions of mobile sensors by Tython [12]. Then we evaluate the performance of our algorithm in a controlled environment provided by TOSSIM [13], which can simulate irregular radio propagation, radio conflict and lossy network behavior at a high fidelity by its empirical probability bit error model. Based on this prototype system and simulation platform, we evaluate the path quality, which shows the effectiveness of our algorithm in producing smoothed path, in tolerating collision with obstacles and in reducing the chances of out-of-field and path oscillation.

The rest of the paper is organized as follows. In section II, we highlight the related work and introduce a two layer architecture for path planning. In section III, we formulate the location-free robot navigation problem and explain an inspiring scenario used throughout the paper. Section IV and V separately present the planning part and the navigation part of our two layer algorithm. Section VI shows the simulation results and section VII concludes our work.

II. Related Work

Path planning is an extensively studied topic in the field of robot motion planning. Proposed algorithms can be broadly grouped into two categories [14]: local and global approaches. Local planning considers only local information about the surrounding, obtained from the sensors carried by robots (e.g., APF [6], VFH+). Global planning is based on the global knowledge of the workspace (e.g., Wavefront [15], A*, roadmap). Generally speaking, local approaches support real time response to obstacles, but are impossible to achieve optimal trajectories and suspectable to trap in local minima. The global approaches can produce optimal and smoothed trajectory, but with slow response to environmental changes.

In the field of traditional robot motion planning, the two-level robot navigation structure (Fig. 1) is a widely-accepted architecture [14]: a global trajectory is calculated in the global planning phase, while in the local obstacle avoidance phase the robot reads its own sensory data and adjusts the local trajectory in a reactive way, which gives the robot the ability to cope with unexpected obstacles. This division into two layers is primarily due to the high computation and communication cost required in most global planning techniques.

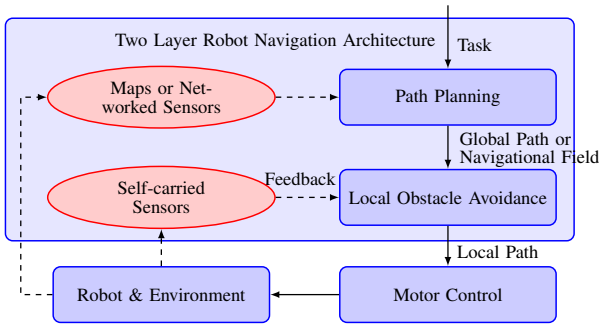


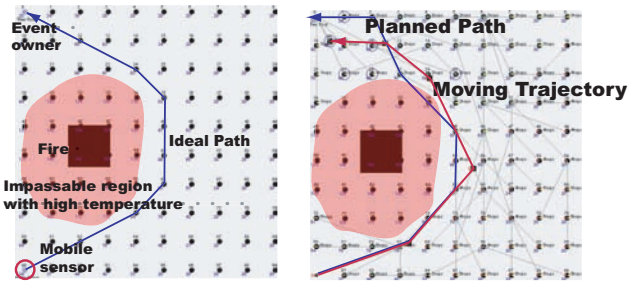
Fig. 1. A two-layer robots navigation system.

The traditional two level architecture needs to be adjusted in the background of hybrid sensor networks, since it is the static sensor network, rather than a map, who gives robots the global knowledge about the whole region and plans the global path accordingly. Compared with the dead maps, the networked static sensors can sketch a real-time picture of the unfamiliar environments. This navigation scheme by sensor networks has been adopted by a recent work [5], which however forget about the self carried sensors in the *Local Obstacle Avoidance* layer.

Another adaptation to the traditional architecture in Fig. 1 is to change the data coupling between the two layers. The traditional coupling by *Global Path* may be unsuitable in hybrid sensor networks, since whenever the old path invalidates (e.g., robots deviate from the old path), a new path should be reconstructed, which is more feasible on a centralized map than within a distributed sensor network due to different cost of path reconstruction. A more appropriate coupling is by a *Navigational Field* covering the entire sensor deployment region, for the following reasons. First, the frequency for the routing path to be invalid can be high in sensor networks, because (1) the routing path is by nature zigzag and may intersects with obstacles, and (2) it can be difficult to control the robots to navigate strictly along the routing path, since this control is implemented via irregular and error-prone wireless communications. Second, a navigational field covering the entire deployment region can give the local obstacle avoidance layer more freedom to steer the robots and avoid unexpected obstacles without the inhibitive cost of recomputing the global path. This freedom is important, since the reliability of navigation assisted by sensor networks can be much lower due to the unreliability of wireless channels.

III. Problem Formulation

A hybrid sensor network comprises of numerous static sensors and a few mobile sensors, with three major assumptions. (1) Different from the static sensors, the mobile sensors can change their geometric locations autonomously. For the sake of simplicity, their mobility platforms are assumed to be free-flying, which explains why the robots can be treated as points. This assumption in future can be relaxed to be holonomic or nonholonomic mobility platform. (2) Although the nodes have different mobilities, their wireless communication components are homogeneous, i.e. the same MAC protocol over the same type of radio antennas configured to the same power level. In our experiments, the adopted wireless module is the TinyOS active message layer [11] over a CC2420 RF transceiver. (3) Robots are assumed to be able to detect the Angle-of-Arrival (AoA) of incoming radio signals, by amplitude anisotropy or phase interferometry. Radio AoA devices are recognized to be too expensive to deploy on the massive static sensors, but they're affordable to equip on a few robots. Fig. 2(a) illustrates such a hybrid sensor network with a grid arrangement of a ten feet spacing (the simulation section presents other network configurations, like random sensor distribution and sparse networks). The exemplified network has only one mobile sensor located at the left bottom corner.



(a) The mission and the ideal path (b) Our trajectory and routing path.

Fig. 2. A rescue mission carried by a hybrid network in an on-fire building.

For a mobile sensor, its free moving space is not the entire region covered by the network, due to the existence of obstacles. The static sensors know the presence of obstacles in their own vicinities, by checking their ADC readings. In our scenario, the obstacles are regions with high temperature, which may paralyze the function of a mobile sensor. A static sensor can measure the temperature and tell whether it's safe for robots by comparing its reading with a predefined threshold. Fig. 2(a) has only one obstacle colored by red, where the temperatures are above a threshold of 100°C.

When an event of interest occurs within the network, static sensors can detect its occurrence and forward this message to robots by some protocol. Mobile sensors then navigate to the phenomenon region to provide actuation, which is the basic motivation of hybrid sensor networks. This complex conceptual process includes event detection, event identification, event notification and robot navigation. Event detection and identification are application specific issues and beyond the scope of this paper. For simplicity, we assume only one static sensor is the owner of an event, whom is probably chosen by a small scale leader election held in the phenomenon region.

The objective of this paper is to design effective algorithms (1) for the static sensors to collaboratively conduct path planning during the event notification phase initiated by the event owner, and afterwards (2) for the mobile sensor to develop a sense of direction and navigate accordingly, until it establishes a direct radio link with the event owner. The constraints for this path planning and robot navigation problem include: (1) final navigation path for the robot should be collision-free for its safety; (2) static sensors do not know their geometric locations; (3) the radio model for sensors, both static and mobile, is not unit disk based but follows the empirical probabilistic bit error model in TOSSIM, which permits irregular radio propagation, packet loss and radio conflicts. Although the above problem definition includes only one event owner and one robot, it can be easily extended to multiple owners and multiple robots, by incorporating the semantics of *Anycast*.

The following two sections present a reliable solution for this navigation problem in wireless sensor networks, comprising of two consecutive phases: a path planning phase followed by a robot navigation phase. The planning phase corresponds to the *Path Planning* layer in Fig. 1 and is discussed in Section IV. The navigation phase relates to the *Local Obstacle*

Avoidance layer and is explained in Section V.

IV. Path Planning Phase

The path planning phase is a process initiated by the event owner to search for a path to the robot and notify the robot about the occurrence of a certain event. But when this phase completes, not a single path but a distributed hop count based artificial potential field (DH-APF) is established, which is essentially a distributed scalar field with only one global minimum at the event owner. The gradients of DH-APF thus, in the subsequent navigation phase, can help the robot develop a sense of direction and navigate to the event owner. This section firstly presents a distributed DH-APF construction algorithm called *distributed wavefront* and later identifies several innate inadequacies of this DH-APF, which inspire us to design a reliable robot navigation algorithm in Section V.

A. Distributed Wavefront Algorithm

The design space for the path planning phase is extremely narrowed by two constraints: distributed computing and location free. The decentralization requirement inhibits the use of centralized search algorithms (e.g., A*), since their coordination among different search branches can incur unbearably high communication cost. The absence of location knowledge bans the use of heuristic search algorithms inspired by the geometric location (e.g., various geometric routing algorithms, GHT, double ruling). The *wavefront* algorithm [15] however can satisfy these two requirements, which is essentially a breadth first path search algorithm resembling the well-known flooding in the field of network routing. A wave is initiated by an event owner and gets propagated among networked static sensors, by which we are able to construct a DH-APF with only one local minimum.

We present a rule-based description of a distributed wavefront algorithm in TABLE I, which is deployed on all static sensors. A typical execution of the rule-based code is as follows. The event occurrence triggers the execution of *Initiate* rule on the sensor who is the event owner. The owner then sends to itself a loop-back message with hopcount -1, which activates the *Push* rule on the owner. The *Push* rule updates the hopcount of the owner as 0 and propagates that number to its neighborhood by rule *Propagate*. Upon the receiving of hopcount 0 from the owner, the neighbors of the owner schedule the execution of their *Push* rules. The process repeats and it terminates when the hopcounts of all static sensors are configured to their proper values (see the *Ignore* rule). However, the sensors in impassable regions is excluded from the above DH-APF construction process, since sensors there by applying the *Impassable* rule disable all their activities. But we do not assume that our static sensors in impassable regions with high temperature in the scenario can endure the heat there, because (1) sensors that are destroyed by the heat can not involve in the DH-APF construction, and (2) sensors still alive voluntarily exclude themselves from the process and pretend to be destroyed.

TABLE I
Distributed Wavefront Algorithm for Static Sensors

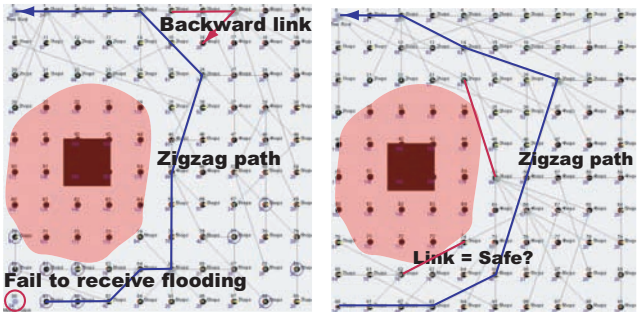
```
# state definitions for a static sensor
integer hopcount := INFINITY.  # my hopcount

# 'defrule' marks the beginning of a definition, while '.' tells the end.
# ':' separates definition name and body.
# '⇒' is a mapping from condition to a sequence of actions.
# ';' indicates a sequential execution between two actions.
# bold letters emphasize events or actions that can signal events.
# rule definitions for a static sensor
defrule Impassable :  # if impassable, it shields all rules that follows
        my vicinity is not suitable for passage ⇒ do nothing.
defrule Initiate :
        become an event owner ⇒ send −1 hopcount to myself.
defrule Ignore :
        receive a hopcount no more than one hop different from mine
        ⇒ do nothing.
defrule Push :
        receive a hopcount at least two hops smaller than mine ⇒
        update my hopcount as the received hopcount plus one;
        activate rule Propagate.
defrule Propagate :
        choose a backoff (half of window plus jitter);
        send my hopcount to vicinity;
        choose a backoff (half of window plus jitter);
        resend my hopcount to vicinity.  # increase comm reliability
```

B. Algorithm Evaluation and Enhancement

The distributed wavefront algorithm establishes a DH-APF covering the obstacle-free region and with a global minimum at the event owner. One such field is shown by Fig. 3(a), in which the three LEDs of each sensor display the lowest three bits of that sensor's hopcount and each gray arrow outgoing from a sensor indicates from which neighbor that sensor updates its hopcount to the current value. However, the DH-APF established by networked static sensors is far from the ideal APF constructed from map, according to our experiments. We identify several problems of DH-APF: (1) Zigzag Planned Path; (2) Link $\neq$ Safe - the possibility that radio links intersect with obstacles; (3) Backward Link; (4) Coverage problem - flooding fails to cover all sensors and the robot may fail to hear about the flooded event. Although these problems are largely inevitable due to the unreliability of wireless links and the sparse distribution of static sensors, we recommend an enhancement by the *Pull* rule to mitigate problem (3)&(4) at the end of this subsection.

The Zigzag Planned Path problem is due to the much lower resolution of sensor distribution than grids in a map and the much longer connectivity range of sensors than the distance between neighboring map grids. Traditionally, the strength of globally planned path based on maps is its promise to (1) generate optimized global pathes, (2) avoid trap in local minima and (3) construct smoothed pathes. When the planning is changed to be conducted by sensor networks, the advantages



(a) Distributed wavefront. (b) Distributed wavefront with pull.

Fig. 3. The enhancement to the generation of DH-APF by the pull rule.

(1) and (2) are partially retained but the third one is sacrificed that will lead to a Zigzag planned path. This low quality zigzag path is one of the reasons that we are reluctant to restrict the robots to navigate along the path planned by sensor networks.

The Link $\neq$ Safe problem may mislead the robots to collide with obstacles, if they follows those dangerous links. It is a fundamental assumption beneath grid based motion planning algorithms [14], that the link connecting two neighboring grids lies in free space if the two grids are in free space. This assumption however may not hold for the sensor networks if we simply regard static sensors as grids. This link $\neq$ Safe problem is one of the proofs that navigation guided by static sensors may not be collision-free and reliable obstacle avoidance assisted by robot sensor readings is necessary.

The Backward Link is a well-known problem of the spanning tree built from flooding, which can potentially mislead a robot to move to a wrong direction. We recommend an additional *Pull* rule (see TABLE II) to reduce the possibility of backward links. When the end node of a backward link propagate its own hopcount by *Push* rule, some neighbor may detect the existence of the backward link, by finding the propagated hopcount is at least two hop larger than the one owned by itself. The detector then correct the backward link by propagating its hopcount and making the end node of the backward link point to itself. The effectiveness of the *Pull* rule to reduce backward links is illustrated in Fig. 3(b). Another use of the *Pull* rule is in Section V for the robot to query its neighbors for their hopcount. When the robot broadcasts a hopcount with value INFINITY, the *Pull* rule of its neighboring sensors will be activated to schedule a propagation of their own hopcounts in near future.

The Coverage problem, similar to the Backward Link problem, is caused by fault-prone radio links. This problem of how to disseminate events reliably to robots has already been solved by adopting the eventual consistency semantics in Trickle [16], which in fact also adopts the *Pull* rule. As a summary, the *Pull* rule is a MUST for three reasons: reliable event dissemination, backward links mitigation and neighborhood querying by robots. The additional energy consumption by adopting the *Pull* rule is inevitable.

TABLE II
ADDING THE PULL RULE

defrule Pull : # increase the chance of propagation of smaller hopcount **receive** a hopcount at least two hops larger than mine $\Rightarrow$ **activate** rule Propagate.

V. ROBOT NAVIGATION PHASE

For the robot navigation phase, we present an algorithm to give the robots a sense of direction based on the reversed gradients of DH-APF, which are calculated by only local communications between an robot and its immediate neighbors. Here, we assume the presence of radio AoA devices on robots to eliminate the need for location knowledge of the neighbors. However, we argue that robot navigation purely by reversed gradients is unsafe and it should be combined with the agile sensing ability of the robot's sensors. Additionally, we compare the performance of navigation field with that of navigation band [5] and show the effectiveness of navigational fields in tolerating deviation and mitigating oscillations.

A. Robot Navigation guided by Reversed Field Gradients

The purpose of this navigation phase is to generate a collision-free moving trajectory starting from the robot's initial position and ending at a point within the one hop range of the event owner. Fig. 2(b) illustrates one such trajectory in our exemplified scenario where the planned path may pass through a dangerous area but the final moving trajectory of the robot remains to be safe. This navigation is a step-by-step process with each step as a line segment. At the beginning of each step, the robot should determines this step's moving direction by querying neighboring static sensors about their hop counts. The step size is an adjustable system parameter.

In TABLE III, we restate the above process without ambiguity by a rule-based language. The activation of rule *StartMission* is a symbol of phase transition from Path Planning to Robot Navigation. The rule *MissionDone* is to terminate this Robot Navigation phase. The rule *MoveAStep*, which recursively activates itself at the end, carries out the step-by-step navigation process. The third line of the *MoveAStep* rule is a request to collect all hopcounts around the robot neighborhood by propagating an INFINITY hopcount to vicinity. The neighbors later respond with their hopcount applying the *Pull* rule in TABLE II. The *CollectGuide* rule listens to and record critical information of these replies, based on which we estimate the reversed DH-APF gradient. In this algorithm description, the robot moves to its next position only following the direction of the estimated reversed DH-APF gradient. This navigation based on reversed gradient can cause trouble, which we shall explain in the next subsection.

Here we describe the algorithm to calculate the reversed gradient $-\vec{\nabla} f$ of the artificial potential field $f(p)$ at the point $p_0 = [x_0, y_0]^T$ where robot locates. If $f(p)$ is a continuous and differentiable scalar field, the required gradient can easily be calculated by $-\vec{\nabla} f|_{p_0} = -\frac{\partial f}{\partial p}|_{p_0}$. However, the DH-APF constructed by static sensors is a discrete scalar

TABLE III
DISTRIBUTED WAVEFRONT ALGORITHM ON A ROBOT

```
# state definitions
import state hopcount defined for a static sensor.
bool   onmission := FALSE.
list   guides of struct <integer nbr, integer hopcount, double aoa >.
struct force < double dx, double dy >.

# rule definitions for a mobile sensor; here '∧' represents logical and
defrule StartMission :
        onmission = FALSE ∧ my hopcount is updated ⇒
        onmission := TRUE;
        wait still until flooding converges;  # guides collection period
        calculate the reversed DH-APF gradient g⃗ from collected guides;
        force := g⃗/|g⃗| · step size.
        activate rule MoveAStep.
defrule MoveAStep :
        move a step driven by the force;
        stop the motor engine; clear the guides list;
        hopcount := INFINITY;  activate rule Propagate;
        stay still for a period;       # guides collection period
        calculate the reversed DH-APF gradient g⃗ from collected guides;
        force := g⃗/|g⃗| · step size;
        if onmission, then activate rule MoveAStep.
defrule MissionDone :
        onmission = TRUE ∧ my hopcount is updated to one ⇒
        onmission := FALSE; stop the motor engine;
        clear the guides list;  hopcount := INFINITY.
defrule CollectGuide :              # guides collection period
        receive a hopcount from the neighor nbr with angle aoa ⇒
        if received hopcount is at least two hop smaller than mine,
        then update my hopcount as the received hopcount plus one;
        append <nbr, hopcount, aoa> to guides list.
import rule Propagate defined for a static sensor.
```

field and the knowledge collected by the robot includes only

$f(p_0)$: potential value at p_0 or hopcount of the robot
$f(p_i)$: potential value at p_i or hopcount of neighbor i
$\overrightarrow{p_0p_i}$: unit vector from p_0 to p_i or the angle-of-arrival of radio from neighbor i to the robot.

Here the p_0, p_i, $\overrightarrow{p_0p_i}$ are all defined in the robot's local coordinate frame, which do not need any translation into the global coordinate frame. The following is the equation to estimate the reversed gradient in the discrete scalar field built by static sensors.

$$-\vec{\nabla} f|_{p_0} \quad = \quad \sum_{f(p_i)<f(p_0)} \overrightarrow{p_0p_i} + \sum_{f(p_i)>f(p_0)} \overrightarrow{p_ip_0} \quad (1)$$

The mathematical meaning of Eq. (1) is the vector of left derivative plus that of right derivative. Its physical meaning is that the *upstream* nodes (neighbors with hopcount smaller than that of the robot) exert attractive forces on the robot, while the *downstream* nodes (with larger hopcount) exert repulsive forces. The direction of the resultant force is treated as the reversed gradient direction. In Fig. 4, the reversed gradient direction is calculated by combining the attractive force from mote 33 with hopcount 1 and the repulsive forces from motes 36 and 38 with hopcount 3. Although these forces direction are known from the robot radio AOA ability, robots do not need to know their magnitude (all unified to one) with the absence of range information.

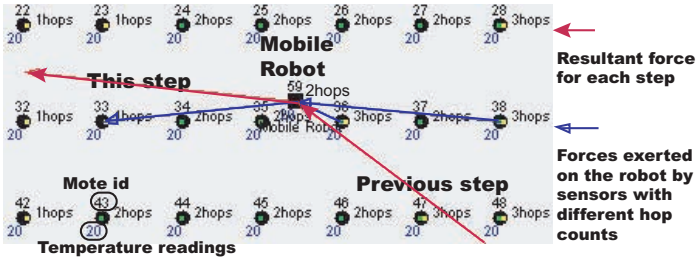


Fig. 4. Deciding the moving direction by the reversed gradients of DH-APF.

B. Algorithm Evaluation and Enhancement

This subsection evaluates the performance of robot navigation based on reversed gradients by both theoretical analysis and experiments. There are three potential problems in our algorithm (also existed in previous navigation algorithms guided by static sensors): (1) possible deviation from planned pathes; (2) possible collision with obstacles; (3) oscillation in narrow passages. After presenting these problems, we also recommend corresponding solutions.

1) Error in reversed gradient estimation and its sources: It's inevitable that an estimated reversed gradient deviates from the real gradient, caused by many factors. For example, even in Fig. 5 where DH-APF is ideal, an obvious deviation can happen between the real gradient $\vec{G}$ and the estimated reversed gradient $\vec{C}$. The $\vec{C}$ is calculated by combining the attractive force $\vec{A}$ from centroid of lower hopcount region and the repulsive force $\vec{B}$ from centroid of higher hopcount region. It's the deviation of $\vec{A}$ and $\vec{B}$ from $\vec{G}$ that causes the deviation of $\vec{C}$ from $\vec{G}$. The deviation of $\vec{A}$ and $\vec{B}$ from $\vec{G}$ is due to the radio irregularity [17] of the robot. Therefore, the radio irregularity of robot is one of the reasons for error in reversed gradient estimation. Other factors include the irregular DH-APF, instability in DH-APF (topology changes), nonuniform sensor distribution in robot's neighborhood and message loss when the robot talks with its neighbors. With so many interfering factors, it's difficult to predict and quantify the error in the reversed gradient estimation.

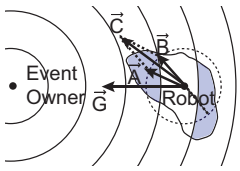


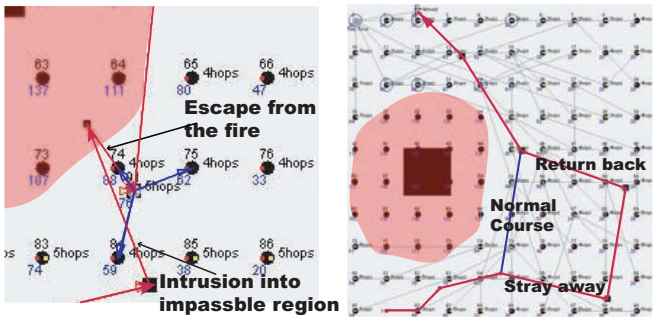
Fig. 5. Error in reversed gradient estimation with isotropic DH-APF.

An unexpected observation about the reversed gradient estimation is that it's more appropriate to revise Eq (1) by Eq (2) adopting only the attractive forces from upstream nodes, which are more robust against network topology change and

more accurate when the robots are turning around corners.

$$-\vec{\nabla} f|_{p_0} \quad = \quad \sum_{f(p_i)<f(p_0)} \overrightarrow{p_0 p_i} \qquad (2)$$

2) Collision problem, deviation problem and their solutions: Caused by the error in the reversed gradient estimation, frequently the robot collides with obstacles and occasionally the robot deviates from the planned path, as demonstrated by our experiments. The collision, or intrusion into the high temperature region in Fig. 6(a), may paralyze the function of robots and endanger the overall task completion. The deviation, like in Fig. 6(b), not just wastes robot's energy but also may lead the robot out of the deployment region, where it's impossible for the robot to seek guidance from static sensors. The possibility of moving out of navigation field is much higher for the navigation band based algorithm, in which only sensors along the planned path can provide guidance. We name this problem as the "out-of-field" problem. However, it is difficult to qualify the possibilities of the two exceptional cases - collision and out-of-field, mainly because of the difficulty in quantifying the error in the reversed gradient estimation.



(a) Detect a collision then escape. (b) Stray away then return back.

Fig. 6. Enhance the robustness of navigation to collision and straying away.

We solve the collision problem by exploiting the agile readings from the robot self-carried sensors, which belongs to the *Local Obstacle Avoidance* layer in Fig. 1. This solution needs two devices on a robot: a temperature sensor and a heat source detector with only coarse precision. As indicated by the *Escape* rule in TABLE IV, the robot detects the collision by comparing current temperature reading with a predefined alert threshold. If it's higher than the alert threshold, the *Escape* rule constructs a repulsive force to avoid the collision. The direction of the repulsive force escaping from the fire is retrieved from the heat source detector and its magnitude is the adjustable escaping step size. The Fig. 6(a) illustrates the working of the *Escape* rule with alert threshold is set to 100 degree and escaping step configured to 10. The robot detects the collision happened by finding its temperature reading is above 100 degree. Then it moves out of the impassable region following the direction away from the fire and with step size 10. After the evacuation, the robot reissues the request for guidance within its neighborhood and continues its mission. Although in Fig. 6(a) the collision actually happens, potential collision can be avoided by changing the behavior pattern from collide-then-escape to predict-collision-then-escape. The

TABLE IV
ADDING THE ESCAPE RULE

defrule Escape :
sensed temperature **is** above the alert threshold $\Rightarrow$
construct a vector $\vec{e}$ escaping from the fire detected by a coarse heat source detector;
force := $\vec{e}/\|\vec{e}\| \cdot$ escaping step size.

collision prediction can be implemented in our scenario by configuring an alert threshold lower than the temperature threshold of impassable region, like 80 degree. The assumed availableness of temperature reading and heat source information to robots can be abstracted as the robots' local obstacle avoidance abilities. The assumption of this ability is also valid in other scenarios, e.g. the wall is the obstacle and the robot has the ultrasound based obstacle sensing ability.

We mitigate the out-of-field problem by expanding the narrow navigation band along planned path to the broad DH-APF based navigation field covering the entire deployment region. Therefore, in Fig. 6(b), even when the robot strays away to the right bottom corner, far away from the the planned route. Still, it can retrieve the reversed field gradient from surrounding and get back to normal course later. We admit that with our solution it is still possible for the occurrence of out-of-field event, e.g. the robot again makes wrong decision about next step moving direction at the right bottom corner. Actually, we doubt the existence of an ultimate solution that robots never get lost. At least, our solution gives the sensor network deployer a method to reduce the out-of-field possibility by throwing more sensors and broaden the deployment region.

3) Motion oscillation problem and its solutions: Another advantage of replacing the navigation bands by our DH-APF based navigation fields is to mitigate the oscillation problem in narrow passages. We observe that when the robot moves in narrow corridors, it often encounters a path oscillation problem, as illustrated by Fig. 7. When the robot locates near the center of the corridor, it tends to move by a smoothed trajectory, represented by a blue line in graph. However, when there is a disturbance that the robot locates near the boundary of the corridor, its moving path oscillates along the blue smoothed path. In fact, [5] naturally creates such a corridor by building a navigation band along the planned path and the oscillation in robots' motion can be anticipated. The existence of oscillation problem get verified by our experiment in Fig. 8(a). Here, the initial disturbance required by oscillations is introduced by making the robot to turn around a corner.

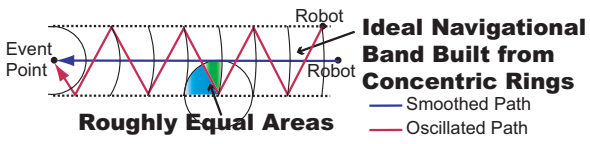
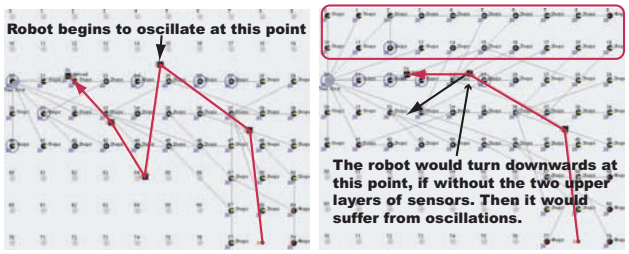


Fig. 7. Model of oscillations in a narrow passage or a navigational band.

The oscillation problem in fact is inevitable for artificial potential field based methods as argued by [6]. We however

can mitigate it here by throwing more sensors and expanding the corridors. In Fig. 8(b), with the two layers at the top which provides additional attractive forces, the motion of the robot no longer oscillates. It turns around the corner smoothly, which shortens the time to arrive at the target. Another cheaper mitigation than the corridor expansion is reducing the step size when detecting a dramatic change in the sequence of reversed gradient estimates. When there is only slight changes in the gradient estimates sequence, we can enlarge the step size to reduce the interactions between robots and static sensors, thus save the precious energies of static sensors and increase the robots' moving speed. A thorough investigation into the energy features of this dynamic step size scheme is our future works.



(a) Unstable motions in narrow corridors. (b) Stable motions in broad regions.

Fig. 8. Oscillation in narrow passages and a possible mitigation.

4) Conclusion: The robot navigation guided by networked static sensors is unreliable, which requires careful engineering efforts. This unreliability mainly originates from the irregular, fault-prone nature of wireless communication and can be worsen by sparse and nonuniform sensor distributions. Our proposed solutions here are to increase robot navigation reliability by exploiting robot self-carried sensors to grantee the collision-free property, and by throwing more sensors to cover a broader region to mitigate the out-of-field problem and the path oscillation problem, especially when the robot turns around the corners. With these efforts on navigation reliability, it's possible for our algorithm to configure a large step size, which reduces the interactions with static sensors, saves sensors' energies and increases the speed of robots. This large step size is impossible for other robot navigation algorithms, since the large step size increases the possibility of out-of-band, oscillation and collision, which can not be well handled by them.

VI. Simulations

A. Experiment Setup

We setup our simulation platform as described below. The path planning algorithm in TABLE I&II is developed by nesC over TinyOS [11]. The robot navigation algorithm in TABLE III&IV is implemented by a mixture of nesC and Tython scripts [12]. We simulate the unreliable radio links by TOSSIM's empirical bit error model [13]. The sensors are placed either in regular grids or in disturbed grids. Although the grid spacing is fixed throughout our experiments, we still have control over the network density or the average node degree by adjusting the radio transmission range. Obstacles in the sensor deployment region can be simulated by one of the two ways: (1) turn off motes in the obstacle regions; (2) create a Tython SimObject with a high temperature attribute.

We use the follow notations for our experiments:

$\mathcal{D}$: sensor distribution density $\frac{100}{100 \cdot 100 \text{ft}^2} = 0.01 \text{sensor/ft}^2$

$\mathcal{R}$: average symmetric communication range $\approx$ 20ft

k : range scaling factor

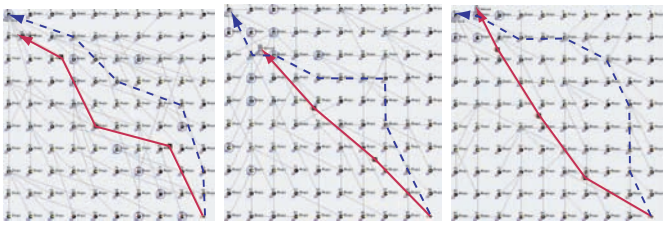
r : average symmetric communication range $\frac{R}{k}$

d : average node degree in the symmetric topology $\pi r^2 \mathcal{D}$

B. Experiment on Trajectory Smoothness

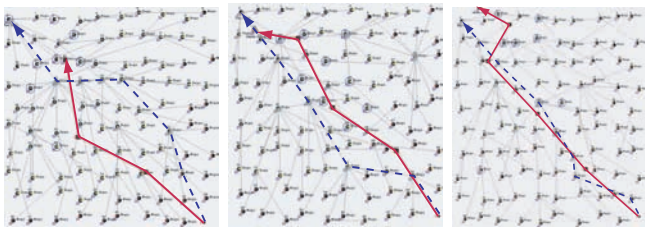
Let l_t be the length of robot trajectory generated by our reliable robot navigation algorithm and l_r be the length of a reversed routing path. Our experiments, by comparing l_t with l_r, are to demonstrate our algorithm can produce smoother trajectories than the shortest routing path and it's safer to function in twisted corridors environments. The reason why we choose the shortest routing path for comparison is that many algorithms are proposed to steer the robot along it, though they may not actually achieve the same efficiency with routing path.

In Fig. 9∼12, robot trajectories are represented by red polylines, while shortest routing paths are represented by blue dotted polylines. The red robot trajectories do not need to end precisely at the event owner, since the robot stop its engine once it arrives within the one hop range of the event owner. Our algorithm has been tested in various network configurations. The primary network parameter we varied is the radio range r by adjusting the scaling factor k. The functional relation between r and k can be found in Subsection VI-A. Other factors include the sensor distribution pattern (regular grid topology or disturbed grid topology) and deployment region (with or without obstacles). In experiments, the function adopted to estimate the reversed field gradients is Eq. (2).



(a) $k = 1, d = 12.56$. (b) $k = 1.15, d = 10.9$. (c) $k = 1.3, d = 7.44$.

Fig. 9. A comparison in rectangular fields with disturbed grid placement.

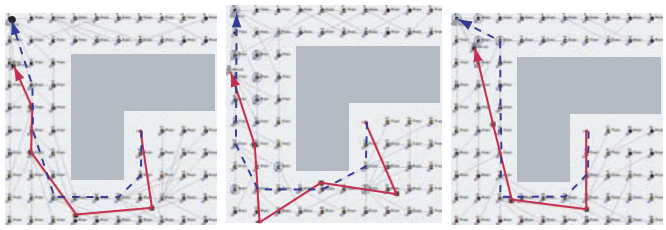


(a) k=1, d=12.56. (b) k=1.15, d=10.9. (c) k=1.3, d=7.44.

Fig. 10. A comparison in rectangular fields with disturbed grid placement.

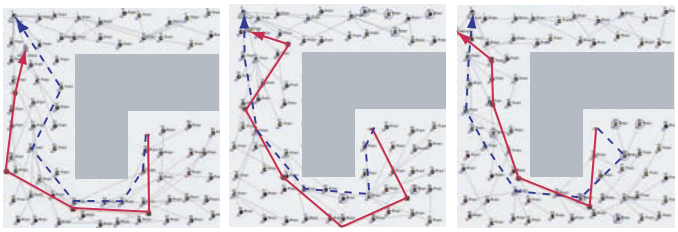
In Fig. 9, the length of multihop routing paths in a regular grid topology is sensitive to the change in sensor density. However, no obvious variations can be witnessed for the length of red trajectories generated by our algorithm. Compared with the shortest routing paths, our trajectories are constantly shorter and more smoothed. This advantage originates from Eq. (2), because in the routing path, only one node is chosen from a pack of upstream sensors as the next hop node, and however, in our robot trajectory, the next step moving direction is determined by "averaging" the attractive forces by all upstream sensors. This advantage can also be witnessed in disturbed grid topologies in Fig. 10, i.e., although the routing paths become smoother in disturbed grid topologies than in regular grid topologies, our trajectories still exhibit better quality. Another observation from Fig. 9&10 is that our trajectories is not sensitive to the change in network densities. This is probably due to the small amount of data exchanges required by our algorithm. Therefore, a denser network with symmetric node degree 12.56 does not necessarily mean much heavier message loss (due to radio conflicts), and a sparse network with symmetric node degree 7.44 is sufficient for the robot to make a good decision about the next step moving directions.

Fig. 11&12 make similar comparisons in twisted corridors. The shortest routing paths have very short moving distance, which are confined by the corridors and are prone to stick to the corners when turning around them. As a comparison, robots applying our algorithm tend to keep a descent distance away from the obstacles, since the lower-hop-count upstream nodes that do not stick to obstacles also exert their attractive forces over the robot. Therefore, though our trajectories are a little longer than the routing paths in twisted corridors, they provide higher safety.



(a) k=1, d=12.56. (b) k=1.15, d=10.9. (c) k=1.3, d=7.44.

Fig. 11. A comparison in twisted corridors with regular grid placement.



(a) k=1, d=12.56. (b) k=1.15, d=10.9. (c) k=1.3, d=7.44.

Fig. 12. A comparison in twisted corridors with disturbed grid placement.

VII. Conclusion

The algorithm presented in this paper provides a solution to guide a mobile sensor to the region of phenomenon in a reliable way. This reliability is achieved by (1) avoiding the local trap by exploiting a global DH-APF; (2) solving the problem of deviation from a planned global path; (3) avoiding collision with obstacles; (4) mitigating the motion oscillation problem. Though the proposed algorithm cannot ensure perfect toleration of navigation path deviation and oscillation, it gives insight as how to use the DH-APF to reduce the possibility of out-of-field problem and improve the navigation trajectory efficiency. Besides the reliability features, this algorithm is also energy-efficient in the sense of (1) saving the energy of static sensors by reducing the interaction between robot and them; (2) saving the energy of robot by constructing a smoothed trajectory shorter than or at least comparable to the shortest routing path. The analysis and experiments in the paper are of a high fidelity, which consider practical issues of error-prone wireless commutations with implementations on TinyOS.

Acknowledgment

This work was supported in part by HK RGC PolyU 5322/08E.

References

[1] Q. Li, M. De Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *MobiCom '03*, 2003.
[2] B. Xiao, H. Chen, and S. Zhou, "Distributed localization using a moving beacon in wireless sensor networks," *IEEE TPDS*, vol. 19, no. 5, pp. 587–600, 2008.
[3] B. Xiao, L. Chen, Q. Xiao, and M. Li, "Reliable anchor-based sensor localization in irregular areas," *accepted in IEEE TMC*, 2009.
[4] C. Sharp, S. Schaffert, A. Woo, N. Sastry, C. Karlof, S. Sastry, and D. Culler, "Design and implementation of a sensor network system for vehicle tracking and autonomous interception," in *Wireless Sensor Networks*, 2005, pp. 93–107.
[5] A. Verma, H. Sawant, and J. Tan, "Selection and navigation of mobile sensor nodes using a sensor network," in *PerCom*, 2005, pp. 41–50.
[6] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *IEEE Int. Conf. Robotics and Automation*, 1991, pp. 1398–1404.
[7] S. Garrido, L. Moreno, D. Blanco, and M. l. Munoz, "Sensor-based global planning for mobile robot navigation," *Robotica*, vol. 25, no. 2, pp. 189–199, 2007.
[8] C. Buragohain, D. Agrawal, and S. Suri, "Distributed navigation algorithms for sensor networks," *INFOCOM*, pp. 1–10, April 2006.
[9] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," in *MobiCom*, 2001.
[10] G. Veltri, Q. Huang, G. Qu, and M. Potkonjak, "Minimal and maximal exposure path algorithms for wireless embedded sensor networks," in *SenSys*. ACM, 2003, pp. 40–50.
[11] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler, "The emergence of networking abstractions and techniques in TinyOS," in *NSDI'04*, 2004, pp. 1–1.
[12] M. Demmer, P. Levis, A. Joki, E. Brewer, and D. Culler, "Tython: A dynamic simulation environment for sensor networks," EECS Department, UCBerkeley, Tech. Rep. UCB/CSD-05-1372, 2005.
[13] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: accurate and scalable simulation of entire TinyOS applications," in *SenSys'03*.
[14] J.-C. Latombe, *Robot Motion Planning*.
[15] J. Barraquand, B. Langlois, and J. C. Latombe, "Numerical potential field techniques for robot path planning," in *Advanced Robotics, 1991*.
[16] P. Levis, E. Brewer, D. Culler, and et al, "The emergence of a networking primitive in wireless sensor networks," *Comm. of the ACM, 2008*.
[17] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher, "Range-free localization and its impact on large scale sensor networks," *Trans. on Embedded Computing Sys.*, vol. 4, no. 4, pp. 877–906, 2005.