Participatory Sensing Algorithms for Mobile Object Discovery in Urban Areas

Harald Weinschrott, Julian Weisser, Frank Dürr, and Kurt Rothermel
Institute of Parallel and Distributed Systems
Universitätsstraße 38, 70569 Stuttgart, Germany
<weinschrott|weissejn|duerr|rothermel>@ipvs.uni-stuttgart.de

***Abstract*—This paper introduces mechanisms for the automated detection of mobile objects in urban areas. Widely available devices such as mobile phones with integrated proximity sensors such as RFID readers or Bluetooth cooperatively perform sensing operations to discover mobile objects. In this paper, we propose a coverage metric for assessing the completeness of sensing that considers spatial and temporal aspects. To maximize coverage while minimizing energy consumption of mobile nodes, we propose both a centralized and a distributed coordination algorithm for selecting nodes that need to sense. Moreover, we present strategies that allow selected nodes to perform efficient sense operations. By extensive simulations, we show that distributed coordination achieves drastic energy savings of up to 63%, while limiting the coverage loss to 13%. Moreover, we show that the centralized algorithm loses less than 1% coverage compared to the maximum possible coverage.**

I. Introduction

The evolution and proliferation of mobile phones in recent years promise to provide multi-sensor platforms for environmental sensing. Today, phones have several integrated sensors such as RFID-readers [1] and GPS [2], and external sensors can be easily connected to the phones [3] using for instance Bluetooth. At the same time, a growing number of people is willing to contribute to so called community-based projects, which for instance create detailed maps of the environment (cf. OpenStreetMap [4]). Mainly, these trends lead to a concentration of research on public urban sensing [2], [5], which aims for environmental sensing with the abundantly available mobile resources in urban areas.

In addition to generating maps of environmental conditions such as noise [6] or air pollution [7], pervasive mobile devices can be used to detect the proximity of mobile objects [8] using proximity sensors based on Bluetooth or RFID technology. Such a mobile sensor network of phones can track mobile objects allowing for applications ranging from lost-and-found scenarios to the tracking of objects such as buses to estimate, for instance, accurate arrival times. Another scenario is the collection of traffic information, e.g., the number of cars on a street segment. Traffic control systems could use this information to reroute traffic or impose temporary speed limits.

The potential of public sensing for these tracking applications comes from the large number of people that cover large urban areas while carrying their mobile phones. Such mobile sensors can achieve a coverage beyond that of the same number of stationary sensor nodes [9]. However, the use of these mobile sensors introduces two main challenges. Due to uncontrolled node mobility, no guarantees about coverage and, therefore, completeness of the search are possible. Therefore, the first challenge is to determine the quality of tracking, i.e., the completeness in terms of coverage of the search area. Second, sensing may not interfere with normal operation of mobile nodes. In essence, this requires efficiency in terms of energy consumption, i.e., sensing and communication of mobile nodes needs to be limited. In our previous works [10], [11], we have presented coordination algorithms for the efficient sensing of stationary phenomena. However, for mobile object tracking, coverage metrics need to be redefined, and completely new coordination algorithms need to be designed.

The main contribution of this paper are concepts and algorithms for the coverage-aware automated tracking of mobile objects in urban areas. Although coverage metrics are available for wireless sensor networks and mobile sensors moving along roads of urban areas, these metrics are not applicable for the proposed scenario of mobile object tracking. Therefore, our contribution comprises coverage metrics that integrate spatial and temporal coverage to reflect the requirements of tracking applications. Moreover, this paper presents a centralized and a distributed algorithm for coordinating sensing of mobile nodes to achieve the maximum possible coverage, while minimizing the energy consumption of nodes. Finally, using extensive simulations, we show the effectiveness and efficiency of these algorithms.

In the following section we present related work (Section II). Then we introduce our system model (Section III), before we discuss quality metrics (Section IV). Afterwards, we present the algorithms (Section V) for coordinated sensing. Then, we present the evaluation setup and results (Section VI). Finally, we conclude this paper and give a brief outlook on future work (Section VII).

II. Related Work

In the research field of public sensing, the focus lies on the mapping of environmental phenomena. Approaches such as [3], [12] use instrumented mobile devices to collect shared sensor data. While Rudman et al. [12] attach sensors for monitoring air pollution to a tablet PC, MobGeoSen

Published in Proceedings of the 9th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'11), 2011, pp. 128-135. Seattle, USA, March 2011.

http://dx.doi.org/10.1109/PERCOM.2011.5767577

[3] relies on mobile phones with integrated sensors that can be carried around by a large number of people. As [13] shows, tracked mobile sensors allow for a fine-grained mapping of environmental effects. Approaches such as [11], [14] consider quality aspects of this mapping. In our previous work [10], [11], we presented distributed coordination algorithms to efficiently achieve quality requirements for mapping stationary phenomena. Similar to us, [15] and [8] use instrumented mobile devices to find objects, and their scoped query mechanisms complement our work in that they allow to select regions for sensing. However, our paper is the first to propose algorithms for the efficient detection of mobile objects in public sensing.

In the field of sensor networks, coverage metrics [16] and mechanisms [17] for tracking or detecting objects have been actively researched. However, research in the field of sensor networks focuses mainly on the autonomous operation in inaccessible areas. Since these scenarios fundamentally differ from our scenario of mobile sensors moving uncontrolled in urban areas, available coverage metrics that assume stationary or controlled sensors moving in the plane cannot be applied to our scenario. To the best of our knowledge, this is the first work to propose algorithms for quality-aware and energy-efficient detection of mobile objects in urban areas.

Our work is complemented by AnonySense [18], which proposes a framework for preserving privacy of participating users in urban sensing scenarios.

III. System model

Our system consists of a *reader network*, a *central server*, and *mobile objects*. Readers form a network and cooperatively perform sensing operations to detect the mobile objects. Next, we describe these components and the underlying assumptions.

A central server is associated to a specific geographic region (service area). The reader network consists of mobile nodes moving in the service area. Their movement is uncontrollable but restricted to the street network. The maximum node speed is v_{maxn}. Nodes have integrated proximity sensors (RFID, Bluetooth, or ZigBee) for detecting mobile objects. Moreover they have a GPS receiver, which allows to derive trajectory information. For ad-hoc internode communication, nodes have a wireless communication interface, e.g., 802.11bg, with transmission range r_{tx}. In Section V-B we will show how the ad-hoc network is used for the distributed detection of objects. In addition, we assume that nodes have WWAN connectivity (GPRS or UMTS). In particular, WWAN is used for transmitting readings from nodes to the central server.

We assume the proximity sensors integrated in the mobile nodes can identify mobile objects with a specific probability p_{detect} if the object is within the maximum detection range r_{detect}. In case of an RFID reader or Bluetooth device, r_{detect} is several meters and documented in a sensor's data sheet. Finally, we assume a sensor to instantly report a reading when queried.

Similarly to mobile nodes, mobile objects O_i move uncontrollable on the street network. Their speed is limited to v_{maxo}. Each object has a unique identifier, for instance, Bluetooth MAC address or ID of RFID tag, which is reported by the proximity sensor of a mobile node.

The street network is modeled as a graph G. Edges represent segments S of a street. Their weight denotes the length of the respective segment. Its width is assumed to be smaller than r_{detect}. If this is not the case, e.g., on broad streets, a street is modeled as multiple parallel segments representing different lanes. Note that this approach is not suited for very short rage sensors such as NFC readers. Readers and objects enter or leave the street network or transit to a different segment only at a vertex. Such vertexes may be junctions, where objects transit to another segment, or doorways, were objects enter or leave a building.

IV. Coverage-aware Query Interface

We want to allow applications to query for objects, and refer to it as object query Q. The query takes an object O_i, and the maximum acceptable search time t_{max}. It returns the position $pos(O_i)$ of the detected object, and a value C_{avg} which is the average of the values $C_i \in [0, 100]$ denoting the percentage of the covered search space for each segment of the service area (cf. Section IV-A).

$$\text{Object Query:} \quad Q\left(O_i, t_{max}\right) \rightarrow \left(pos(O_i), C_{avg}\right) \qquad (1)$$

When the object is not found, $pos(O_i)$ is nil. Then, C_{avg} indicates the possibility of a false negative, i.e., the probability that the object is in the service area but was not found because of incomplete coverage.

A. Quality Metric

To assess the quality of the search result, the coverage metric must consider the temporal aspect in addition to the spatial aspect of sensing. This is in contrast to spatial coverage metrics used for stationary phenomena [11].

Due to continuous movement of mobile objects, a single reading at a certain point in time also covers a certain area after and also before the reading time. Figure 1a shows a tx-diagram, where the x-axis shows the one-dimensional spatial position on a segment, while the t-axis represents time. A reading acquired at position x_{read} spatially covers the surrounding up to a distance of r_{detect} at t_{read}, i.e., it covers the interval $I = [x_{read} - r_{detect}, x_{read} + r_{detect}]$. To derive the coverage of this reading at other points in time, we consider a mobile object that is at the time of the reading just outside I moving at maximum speed v_{maxo} (worst case). As it passes through I, the covered area shrinks according to v_{maxo}. At time $t_2 = t_{read} + r_{detect}/v_{maxo}$ the coverage of the reading is zero. Similarly to the covered triangle after t_{read}, a reading also covers a triangle before the reading.

To clarify this, we consider a mobile object at x_{read} at time $t_1 = t_{read} - r_{detect}/v_{maxo}$. This object needs to move with maximum speed v_{maxo} to leave I before the reading occurs and it would be detected.

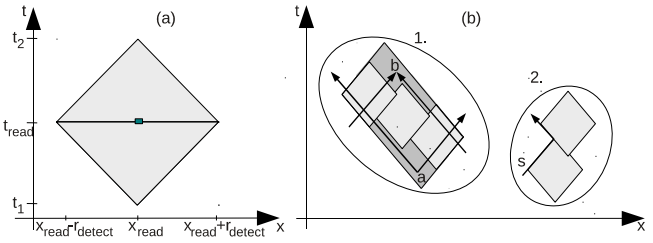

Figure 1. Coverage of a single reader

In general, a reading covers a rhomboid area R. A point P of the tx-plane is covered by R if $P \in R$. Figure 1b illustrates the coverage of multiple readings.

The first group shows the coverage of a reader moving faster than v_{maxo} and acquiring three readings whose rhombi overlap. In this case, additional past and future areas (dark gray areas in Figure 1b) are covered. To understand why the past gray area is covered, consider an object at position a. Every possible trajectory from this location intersects with a coverage rhombi, i.e., the mobile object is detected by the reading of the rhombus. Analogous, the future gray area can be explained. Any mobile object that reaches b must have passed through at least one rhombus. This is illustrated by the two maximum speed trajectories that lead to b.

The second group consist of two overlapping rhombi. In this case, when a node is slower than v_{maxo}, a mobile object is fast enough to approach a reader, turn, and then withdraw again from it before the reader senses again. Figure 1b visualizes this with the trajectory s of a mobile object.

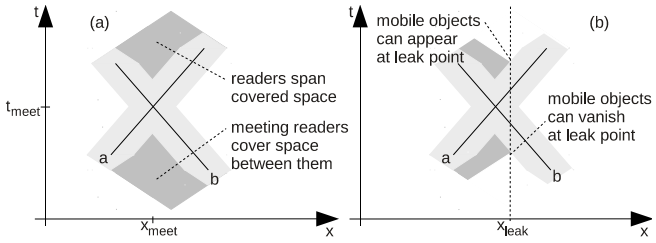

Figure 2. Coverage of multiple readers

This principle can be generalized to the case of readings from multiple readers. For instance, Figure 2a shows the trajectory and the covered area of two readers within the boundaries of a segment. For simplicity, the figure only shows the coverage achieved by continuous reading.

Intuitively, two nodes a, b that move towards each other cover the space between them and, as soon as they meet at time t_{meet}, it can be deduced that the space between them is covered since no object can pass a reader undetected. Furthermore, if these nodes move in opposite directions after the meeting, it can be deduced that the space between them is covered. Figure 2b shows the effect of a leak point, e.g., a crossing where mobile objects enter or leave a segment, on the coverage.

In essence, the coverage principles discussed in this section can be unified as follows. A point P of the tx-plane is covered if at least one of the following conditions hold. First, all valid trajectories (restricted by the maximum object speed v_{maxo}) of objects from P onwards intersect at least one coverage rhombus. Second, all valid trajectories of objects leading to P intersect at least one coverage rhombus.

B. Coverage Metrics

Based on these considerations, we propose two coverage metrics for a segment: The current coverage for a given point in time, and the average coverage during a time period.

Definition 1. *The current coverage $c(t)$ of a segment at a specific point in time t is the ratio of covered segment and segment length.*

Definition 2. *The cumulated coverage C during the time period (t_{min}, t_{max}) is the integral over $c(t)$ normalized by the length of this time span.*

$$C = \frac{1}{t_{max} - t_{min}} \int_{t=t_{min}}^{t_{max}} c(t) \cdot dt \qquad (2)$$

In essence, we evaluate C for a specific segment and time period by cutting the polygons on the respective segment and time period and computing their area. The sum of these areas is divided by the the segment length and the time period.

V. Algorithms for Reader Coordination

In this section, we describe our approach to coordinate mobile readers in order to resolve object queries. With this approach we aim for two goals. First, we want to provide query results with high coverage values to increase the probability of discovering the wanted mobile object.

Second, we want to perform sensing as efficiently as possible. In particular, we want to avoid unnecessary sensing and communication operations that would not increase the coverage, to save energy of the battery-powered nodes.

A straightforward approach would be to let nodes sense continuously while on queried segments. On leaving the segment or when the query period is over, they transmit their readings to a central server. This server collects the readings from all nodes, and computes their coverage. Basically, such an *isolated* approach where nodes sense independently achieves the highest coverage. However, it is likely to produce redundant readings with high node density.

Our goal is to achieve the coverage of the isolated approach, while minimizing the sensing. We propose to reduce sensing in two ways. First, by selecting and coordinating readers such that only essential readers participate

in sensing. For this purpose, we propose two algorithms. The central coordination algorithm is based on a central instance that coordinates sensing of mobile readers (cf. Section V-A). The distributed coordination algorithm is based on the distributed coordination of readers on a segment (cf. Section V-B). To further reduce sensing, we propose sensing strategies in Section V-C that allow, in contrast to continuous sensing, to reduce the sampling rate for sensing.

A. Centralized Coordination Algorithm

The basic idea of this algorithm is to track the position and the sensing state of mobile readers at a central instance. The central instance selects and activates readers based on their current state and the sensing state of their neighbors on the segment. Given a set of reader trajectories, an offline algorithm could decide for each of these, which parts are redundant for sensing. However, due to uncontrollable node movement, it is impossible to prevent all this redundancy in a live algorithm, while assuring the maximum possible coverage. In this section, we aim for a pessimistic algorithm that focuses on maximizing the coverage, i.e., a reader only deactivates sensing if it is within a covered area. First, we present reader states and the corresponding state transitions that occur at meeting points of nodes that allow for identifying nodes in covered areas. Then, we present an efficient update protocol that allows to collect the necessary position information to detect meetings of nodes.

In the following, we assume that a position of a reader on a specific segment is represented by a 1D coordinate. We say position P_L is left of P_R and P_R is right of P_L, if $P_L < P_R$. We distinguish the following reader states.

- **RL-active:** The reader is sensing independently, i.e., it does not span a covered area with another reader. This is the initial state of each reader as it enters a segment.
- **L-active:** An L-active reader extends the left side of a coverage polygon.
- **R-active:** An R-active reader extends the right side of a coverage polygon.
- **inactive:** The reader is inactive.

State transitions occur at meetings of nodes based on reader states. Table I shows all reader combinations at meeting points, and their state transitions. The first column shows the state of the reader from the left and the top row shows the state of the reader from the right. The fields of the table show the new states of the readers. The first state is that of the reader from the left and the second state is that of the reader from the right. Combinations of states that cannot occur in a real scenario are marked accordingly.

In Figure 3a, nodes a and b are RL-active. When they meet at (a, b), they change their states: a to R-active and b to L-active. From the meeting point, these nodes span the covered area between them. Figure 3b shows the case where an RL-active node c from the left meets the L-active node b. After meeting at (c, b), node c enters a covered area and transits to inactive, while b remains L-active. When c leaves the covered area at (c, a), it transits to R-active. At the same time, node a is in a covered area and transits to inactive.

Table I
READER STATE TRANSITIONS

	RL-active	**L-active**	**R-active**	**inactive**
RL-active	R-active L-active	inactive L-active	-	-
L-active	-	-	R-active L-active	inactive L-active
R-active	R-active inactive	inactive inactive	-	-
inactive	-	-	R-active inactive	inactive inactive

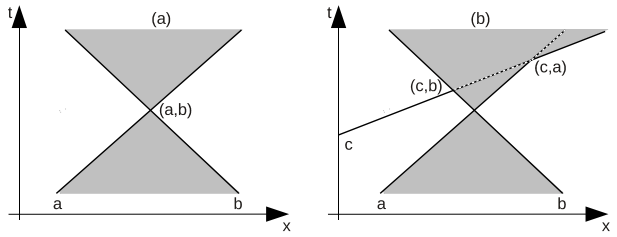

Figure 3. Reader state transitions

Coordination reduces sensing of readers. Additionally, reducing positioning and communication of position fixes to the central server saves energy. To track reader states, the server needs to be informed about their positions. Since it only needs to know where and when nodes meet, their position update frequency can be optimized accordingly.

The idea of this optimization is that nodes may deviate from their predicted movement without updating as long as they move within predefined border lines in the tx-plane to direct neighbors. Assuming constant node speed, the server sets the borders as bisecting lines of the predicted trajectories of neighbors. In addition, segment boundaries are borders.

A node locally determines its positioning rate according to a worst case prediction, i.e., it computes the time to reach a border line at maximum speed v_{maxn}. Note that sensing may have stricter requirements on positioning (cf. Section V-C).

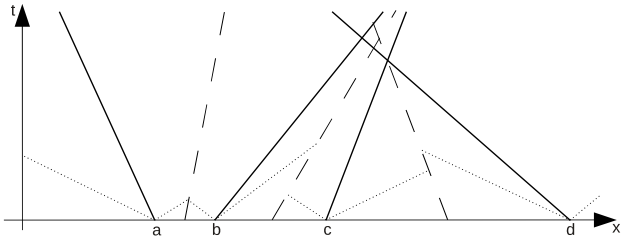

Figure 4. Positioning deactivation

Figure 4 shows four readers and their extrapolated trajec-

tories based on the current speed. Border lines are dashed. A node further delays a position update until it reaches the maximum speed line (dotted lines) of its neighbor. In essence, a node may cross a border until the respective neighbor reaches the border at maximum speed. For instance, c updates only when left of b's maximum speed line.

On receiving an update, the server computes a new border line and notifies the neighbors accordingly. If it detects a meeting of neighbors, it updates their neighbor relations and their state transitions. Note that inactive readers do not need to synchronize with other inactive readers.

B. Distributed Coordination of Reading

Although the pessimistic centralized algorithm is effective, it does not exploit energy efficient local communication over WiFi, which is feasible since coordination only requires communication between neighbor nodes. Therefore, we present a distributed algorithm that is based on reader cooperation in the area of a segment in an ad-hoc network.

In contrast to the centralized algorithm, we aim for an *optimistic* algorithm where nodes locally decide whether they need to sense or not based on local knowledge about the predicted movement and coverage of neighbors. The basic idea of the algorithm is that nodes manage a local view of the node movement on their segment. To achieve this, they broadcast their position and speed in the ad-hoc network if their actual movement deviates from the predicted by more than a threshold. Moreover, nodes notify their neighbors when they enter or leave a segment. Direct neighbors receive these updates and compose them to their local view. To further distribute these updates, nodes include the most recently received updates into their own update messages. Receiving nodes update their local view of a node's trajectory if the received one is more up-to-date.

The basic idea of our approach is to use the predicted trajectories to project the resulting coverage of a group of nodes. If a node can extend their coverage it starts sensing. Otherwise, it deactivates sensing to save energy. If an area can be covered by multiple nodes, rules are required to select the sensing node in order to avoid redundant readings or coverage holes. Thus, we propose an absolute ordering of nodes according to the node ID. In detail, the algorithm is listed in Figure 5.

In the first step, the algorithm iterates over all nodes on the segment and checks incrementally if they extend the coverage. If so, they are added to the list of candidates. Otherwise, they do not need to sense and $sleepUntil$ is set to infinity. Then, the algorithm computes for each candidate the first intersection of its predicted $trajectory$ with the coverage polygon $remain$ resulting from the set of candidates without the node itself ($coverage(\mathbb{C} \setminus N)$). If the node is currently inside $remain$, the intersection I is the point of the tx-plane until which the node can deactivate sensing, i.e., $N.sleepUntil \leftarrow I$. Otherwise, the node needs to sense, and

Require: $\mathbb{L}$ ordered list of nodes
  $\mathbb{C} \leftarrow \emptyset$ // init set of candidate nodes
  for all $N \in \mathbb{L}$ **do**
    if $coverage(\mathbb{C} \cup N) > coverage(\mathbb{C})$ **then**
      $\mathbb{C} \leftarrow \mathbb{C} \cup N$
    else
      $N.sleepUntil \leftarrow \infty$
    end if
  end for
  for all $N \in \mathbb{C}$ **do**
    $Ray\, trajectory \leftarrow N.getProjectedMovement()$
    $Polygon\, remain \leftarrow coverage(\mathbb{C} \setminus N)$
    $Point\, I \leftarrow remain.firstIntersection(trajectory)$
    if $N.getPosition \in remain$ **then**
      $N.sleepUntil \leftarrow I$
    else
      $N.senseUntil \leftarrow I$
    end if
  end for

Figure 5. Distributed Coordination Algorithm

I is the point where it can stop sensing $N.senseUntil \leftarrow I$. This algorithm is executed on receiving an update message.

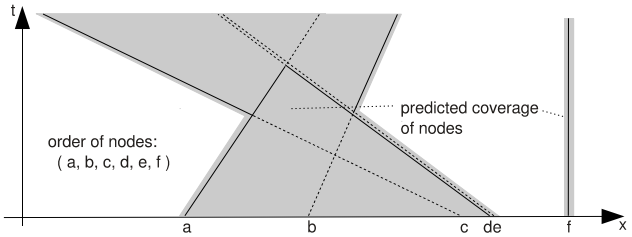

Figure 6. Distributed coordination based on predictions

Figure 6 shows an example. Although, nodes d and e are interchangeable regarding coverage, node e is ruled out due to the ordering of nodes. For the resulting nodes, the algorithm computes the ranges of the trajectories where to sense, e.g., a senses until meeting d.

Deactivating sensing allows for relaxing a node's positioning interval. However, if it enters an uncovered area while positioning is deactivated, coverage is lost. Pessimistically we propose to compute the time period for deactivating positioning as time for a mobile node to reach the coverage boundaries with maximum speed v_{maxn}. A more optimistic strategy is to assume a constant movement, where a node continues to move at its current speed.

C. Efficient Sensing along Trajectories

In the previous sections, we showed how to determine for each node spatial ranges of a segment where to sense. The simplest approach to cover such a range is continuous sensing while a node is within this range.

As Figure 7 shows, a reader can determine a non-continuous but energy efficient sensing interval computed

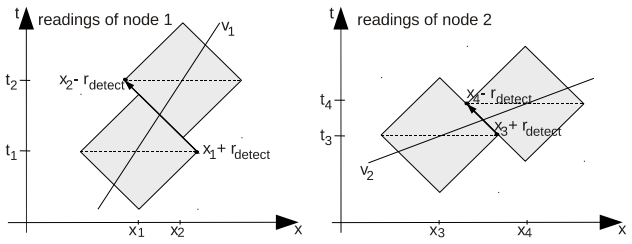

Figure 7. Discrete Sensing Interval

as the time for node and object to cover a distance of twice r_{detect}. However, inaccurate speed predictions may lead to coverage holes, i.e., the coverage of succeeding readings may not overlap. Moreover, since reading is only successful with probability p_{detect}, multiple readings may be needed to actually detect a passing object.

Our goal is to detect mobile objects passing by a sensing reader with a required probability. In essence, this allows for trading of sensing quality, i.e., probability to detect a passing object, and energy cost. This leads to the following equation for computing the time interval between readings.

$$t_{max} = \frac{2r_{detect}}{c_{redundant} \cdot (v_{maxo} + |v_{limit}|)} \tag{3}$$

First, a user specifies the required redundancy $c_{redundant}$ of readings. With p_{detect} close to 1, no redundancy is needed and, therefore, this value is set to 1. If $p_{detect} = 0.9$, a value $c_{redundant} = 2$ is sufficient to detect any passing object with a probability of 0.99. Second, a user specifies the speed v_{limit} a node does not exceed. Optimistically, this is the current speed; pessimistically, it is set to v_{maxn}.

VI. Experimental Setup and Evaluation

This section presents the simulation model and the results of the evaluation of our algorithms. We implemented the algorithms for the network simulator ns-2. In the following we distinguish the following implementations:

- **Centralized:** Pessimistic implementation based on the central coordination algorithm (cf. Section V-A). A central instance coordinates sensing of mobile nodes.
- **Distributed:** Optimistic implementation based on the distributed coordination of nodes (cf. Section V-B), where mobile nodes coordinate sensing in an ad-hoc network. The central instance is only responsible for managing the data read by mobile nodes.
- **Isolated:** Reference implementation where nodes independently sense while on the road network. The Isolated approach presents the worst case for redundant sensing, but also the best case for coverage. This is basically the state of the art [15].

Our implementations use the 802.11 extension of ns-2 with $r_{tx} = 100\,\text{m}$. Nodes and objects move at pedestrian speed (between $0.7\,\text{m/s}$ and $1.8\,\text{m/s}$) according to the UDEL mobility model [19] that is based on surveys from a number of research areas and produces realistic urban traces of pedestrians moving non-uniformly distributed on a street graph. The underlying street graph is a nine block section of Chicago consisting of 93 segments. The influence of node density on our algorithms is evaluated in a wide range of scenarios. The average number of objects at a time on the road network is 150. Movement predictions are based on the assumption of uniform movement at current speed along the current road segment. The position uncertainty is $5\,\text{m}$.

Each edge of the street graph is a road segment. Simulations are performed five times and last for 30 minutes. During that time, nodes try to cover every segment. Nodes have an RFID reader as proximity sensor with $r_{detect} = 5\,\text{m}$, and they use GPRS for communication with the server. Node energy consumption is computed according to Table II.

Table II
Energy Model

Component	Energy [mJ]
GPS [20]	
Position Fix	75
RFID [21]	
Read	80
802.11b at 1 Mbps (broadcast rate) [22]	
Send (1000 Bit)	2
Receive (1000 Bit)	1
GPRS [23]	
Send (1000 Bit)	80
Receive (1000 Bit)	40

A. Effectiveness of Algorithms

In this section, we evaluate the effectiveness in terms of the achieved coverage (cf. Section IV-A). As a performance metric, we compute the average coverage C_{avg} achieved by the mobile nodes per segment during a simulation run.

Figure 8 shows C_{avg} depending on the average number of nodes on the roads. In essence, all three approaches show similarly increasing coverage. Due to redundancy in node movement, C_{avg} scales less than linear with the number of nodes. Isolated achieves the maximum coverage. While Centralized achieves a coverage only slightly lower (about 1%), Distributed further reduces the coverage (at most 13%). This is due to the optimistic design of Distributed that may loose coverage when readers accelerate. However, when the density is low, nodes are more likely to read independently from other nodes and, therefore, the probability of optimistic sensing deactivation is reduced. On the other side, with high reader density, redundancy leads to more optimistic sensing deactivations and, thus, to a higher loss.

Figure 9 shows C_{avg} depending on v_{maxo} with 203 readers. As expected, all three approaches behave similarly. The maximum coverage is achieved for stationary objects. With increasing v_{maxo}, C_{avg} decreases rapidly. However, the

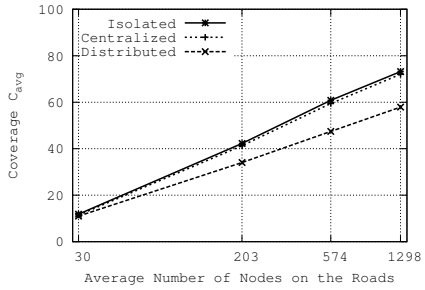

Figure 8. C_{avg} against number of nodes

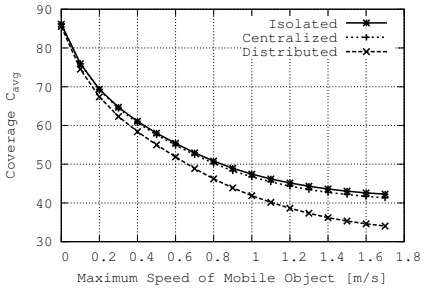

Figure 9. C_{avg} depending on v_{maxo}

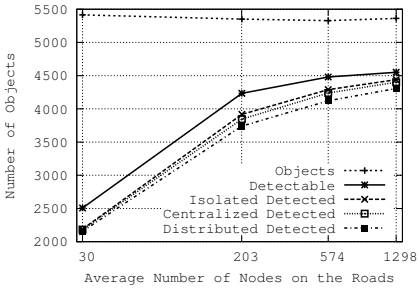

Figure 10. Number of Detections

decrease is limited since covered areas spanned by multiple nodes are independent from v_{maxo}. The explanation for the gap between the three approaches is analogous to Figure 8.

In Figure 10 we compare the overall number of objects, the number of detected objects, and the number of detectable objects. The overall number of objects is computed as the sum of segments entered by objects during the simulation. The number of detectable objects are those which cross at least one coverage polygon. The number of detected objects refers to the results of the three approaches. The average number of nodes is 203. As expected, the overall number is nearly constant at a level of about 5400. The detectable number of objects depends on the number of nodes in the network and, therefore, their coverage. The gap between the actual number of detected objects by Isolated and the number of detectable objects is due to objects that enter or leave a segment within segment boundaries, e.g., at a building entrance. However, the effect of these leak points is limited to 6% in our simulations. Moreover, this figure shows that Centralized and Distributed almost achieve to detect as much objects as Isolated.

In Figure 11, C_{avg} is compared with the detection ratio DR, i.e., the number of detected objects divided by the overall number of detectable objects. Since C_{avg} is based on a worst case mobility assumption for objects, its values are a lower bound for DR. For readability reasons, and because Distributed and Centralized show a similar behavior, we only plot the results of Isolated. Both, C_{avg} and DR similarly depend on the average number of nodes. The higher C_{avg}, the higher is DR. One reason for the gap is that nodes and objects are pedestrians that show effects of correlated group movement. The gap decreases for larger numbers of nodes, when node meetings are likely, and additional coverage is achieved in terms of C_{avg}.

B. Efficiency of Algorithms

Finally, we compare the efficiency in terms of energy consumption (EC). EC is computed as the average energy a node spends per second for positioning, communication, and sensing (cf. Table II). Since Bluetooth discovery and RFID reading consume roughly the same amount of energy, we assume an RFID reader as proximity sensor in the following.

Figure 12 plots the EC depending on the average number of nodes. As expected, Isolated's EC is independent from the number of nodes. In contrast, Centralized and Distributed benefit from increasing node density by preventing unnecessary sensing. Savings are higher with increasing node density, when redundancy of movement increases. Moreover, we observe a large gap between Centralized and Distributed. With 30 nodes on the roads, Centralized's EC even exceeds that of Isolated. One reason is the high power consumption for communication with the server over GPRS. When the node density is low, few redundant sensing are prevented, and even outweighed by the additional communication cost. The second reason for the gap are the design principles of Centralized and Distributed. Centralized is pessimistic and developed to achieve the maximum possible coverage. Distributed is more optimistic, i.e., it may loose coverage. It achieves EC savings of up to 63% compared to Isolated at the cost of only 13% less coverage.

In Figure 13, we compare different sensing strategies (cf. Section V-C). The average number of nodes is 203, and $r_{detect} = 15\,\text{m}$. On the x-axis, from left to right, the optimism of the strategies increases. Most pessimistic is $v_{maxn}, c = 3$. It assumes that the node speed is limited to v_{maxn}, and that the sensing redundancy needs to be set to 3 to achieve the required coverage along a trajectory. More optimistic is $v_{cur}, c = 3$ since it assumes that a node keeps its current speed v_{cur}. Most optimistic is $v_{cur}, c = 1$ without redundancy. As expected, all approaches reduce the EC for higher degrees of optimism (less aggressive sensing). However, the differences between the strategies w.r.t. absolute savings are small, since the cost for positioning cannot be reduced. This suggests that increasing the success probability only slightly increases the EC.

VII. Conclusions

In this paper, we introduced mechanisms for the automated detection of mobile objects in urban areas. Our approach utilizes mobile devices such as smartphones to track mobile objects. We proposed a metric that captures the coverage, and we presented a centralized and a distributed algorithm

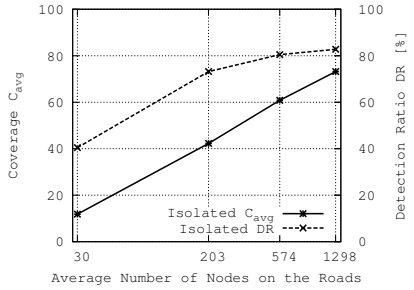

Figure 11. Comparison of C_{avg} and DR

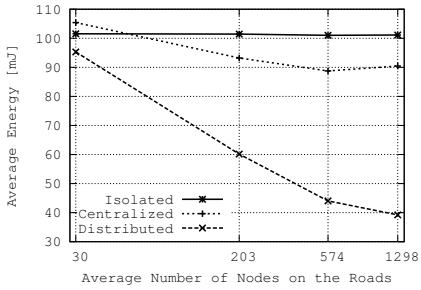

Figure 12. EC against number of nodes

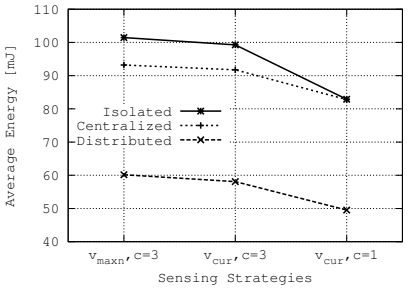

Figure 13. Effect of sensing strategies

to coordinate readers such that redundant sensing is reduced, i.e., energy consumption is reduced. Using simulations, we showed that these algorithms achieve energy savings of up to 63% while reducing coverage by only 13%.

In future work, we plan to investigate a self-tuning approach that automatically adjusts the coverage parameters such that a certain quality of detection is achieved rather than only reporting the achieved coverage. Furthermore, we plan for a real-world evaluation based on Bluetooth.

ACKNOWLEDGEMENTS

This work was partially funded by the German Research Foundation within the Collaborative Research Center 627 (Nexus).

REFERENCES

[1] NFC Forum, Sep. 2010. [Online]. Available: www.nfc-forum.org

[2] D. Cuff, M. Hansen, and J. Kang, "Urban sensing: out of the woods," *Commun. ACM*, vol. 51, no. 3, pp. 24–33, 2008.

[3] E. Kanjo, S. Benford, M. Paxton, A. Chamberlain, D. S. Fraser, D. Woodgate, D. Crellin, and A. Woolard, "Mobgeosen: facilitating personal geosensor data collection and visualization using mobile phones," *Personal Ubiquitous Comput.*, vol. 12, no. 8, pp. 599–607, 2008.

[4] M. M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, Oct. 2008.

[5] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn, "The rise of people-centric sensing," *IEEE Internet Computing*, vol. 12, no. 4, pp. 12–21, 2008.

[6] N. Maisonneuve, M. Stevens, M. E. Niessen, P. Hanappe, and L. Steels, "Citizen noise pollution monitoring," in *dg.o '09: Proc. of the 10th Annual Intl Conf on Digital Government Research*, 2009, pp. 96–103.

[7] S. Kim and E. Paulos, "inair: Measuring and visualizing indoor air quality," in *Ubicomp*, 2009.

[8] C. Frank, P. Bolliger, C. Roduner, and W. Kellerer, "Objects calling home: Locating objects using mobile phones," in *Proceedings of the 5th International Conference on Pervasive Computing (Pervasive 2007)*, Toronto, Canada, 2007.

[9] B. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley, "Mobility improves coverage of sensor networks," in *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, New York, NY, USA, 2005, pp. 300–308.

[10] H. Weinschrott, F. Dürr, and K. Rothermel, "Efficient capturing of environmental data with mobile rfid readers," in *Proceedings of International Conference on Mobile Data Management*, Taipei, Taiwan, 2009.

[11] ——, "Streamshaper: Coordination algorithms for participatory mobile urban sensing," in *Proceedings of the 7th IEEE Intl Conf on Mobile Ad-hoc and Sensor Systems*, San Francisco, CA, USA, 2010.

[12] P. Rudman, S. North, and M. Chalmers, "Mobile pollution mapping in the city," in *Proc. UK-UbiNet workshop on eScience and ubicomp, Edinburgh*, 2005.

[13] A. Steed and R. Milton, "Using tracked mobile sensors to make maps of environmental effects," *Personal Ubiquitous Comput.*, vol. 12, no. 4, pp. 331–342, 2008.

[14] A. Krause, E. Horvitz, A. Kansal, and F. Zhao, "Toward community sensing," in *IPSN '08: Proc of the 7th Intl Conf on Inf. Processing in sensor networks*, Washington, DC, USA, 2008, pp. 481–492.

[15] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *Wireless Communications*, vol. 13, no. 5, pp. 52 –57, 2006.

[16] P. Brass, "Bounds on coverage and target detection capabilities for models of networks of mobile sensors," *ACM Trans. Sen. Netw.*, vol. 3, no. 2, 2007.

[17] P. Medagliani, J. Leguay, V. Gay, M. Lopez-Ramos, and G. Ferrari, "Engineering energy-efficient target detection applications in wireless sensor networks," 2010, pp. 31 –39.

[18] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonysense: Privacyaware people-centric sensing," in *In Proc. ACM 6th Intl Conf. on Mobile Systems, Applications and Services (MOBISYS 08)*, 2008.

[19] J. Kim, V. Sridhara, and S. Bohacek, "Realistic mobility simulation of urban mesh networks," *Ad Hoc Netw.*, vol. 7, no. 2, pp. 411–430, 2009.

[20] Navman, Sep. 2010. [Online]. Available: www.global time.com.cn/attachment/30DataSheet_060616105012.pdf

[21] Skyetek, Sep. 2010. [Online]. Available: www.skyetek.com/Portals/0/Documents/Products/SkyeModule_M9_DataSheet.pdf

[22] Summitdatacom, Sep. 2010. [Online]. Available: www.summitdatacom.com/SDC-CF10G.htm

[23] B. Gedik and L. Liu, "Mobieyes: A distributed location monitoring service using moving location queries," *IEEE Trans. on Mobile Computing*, 5(10), pp. 1384–1402, 2006.