A Sensor Network System for Measuring Traffic in Short-Term Construction Work Zones*

Manohar Bathula[1], Mehrdad Ramezanali[1], Ishu Pradhan[1], Nilesh Patel[2], Joe Gotschall[2], and Nigamanth Sridhar[1]

[1] Electrical & Computer Engineering, Cleveland State University
[2] Civil & Environmental Engineering, Cleveland State University

Abstract. In this paper, we present the design and implementation of a sensor network system for monitoring the flow of traffic through temporary construction work zones. As opposed to long-term work zones which are common on highways, short-term or temporary work zones remain active for a few hours or a few days at most. As such, instrumenting temporary work zones with monitoring equipment similar to those used in long-term work zones is not practical. Yet, these temporary work zones present an important problem in terms of crashes occurring in and around them. Our design for a sensornet-based system for monitoring traffic is (a) inexpensive, (b) rapidly deployable, and (c) requires minimal maintenance. We report on our experiences in building this system, and with testing our system in live work zones in the Cleveland area.

1 Introduction

Construction work zones on roadways are hazardous areas. Motorists are exposed to unfamiliar situations in a normally familiar setting, and such unexpected unfamiliarity could lead drivers to behave in unforeseen ways. Highway work zones are typically long-term installations that are put in place for several weeks, if not months. Such installations are instrumented with a wide variety of sensing and monitoring equipment for observing traffic behavior and for recording unexpected situations. In contrast, utility work commonly takes a few hours, at most a few days to complete. It is not economically feasible to instrument such work zones with the same kinds of equipment used in highway work zones. In fact, in almost every such work zone we see in our neighborhoods, there is *no way* of tracking and monitoring traffic.

Consider the following scenario. The local electric utility company needs to perform maintenance on some street for which they need to encroach into a portion of the street. The utility workers bring their equipment in a utility truck, and before beginning work, they deploy construction cones to demarcate the work area, and to warn drivers. While this level of visual warning works well enough for motorists that are already driving on the street, a motorist who is a mile away, or even just around the corner, typically has no indication of the potential hazard.

* This work was supported in part by grants from the NSF (*CNS-0746632, CAREER* and *REU*), a grant from the US DOT by way of the CSU UTC, and an undergraduate research *Engaged Learning* grant from CSU. Thanks are also due to Area Wide Protective for allowing us to use their live work zones as testbeds.

Further, in the scenario above, if a crash were to occur, the authorities are notified. From interviews of the people present at the time, the causes of the crash may be reconstructed. Such a reconstruction may be flawed: the driver in question may not divulge key errors on their part, witnesses may not have been paying complete attention, etc. Nevertheless, there is at least a record of an incident. There are cases, however, where a motorist may come close to crashing, but is able to recover at the last instant. Such near-crashes are never recorded. These near-incidents are important: the reason that the motorist was put in that situation may have had something to do with the design of the work zone. If such instances were recorded and correlated with work zone design, transportation safety engineers could work on avoiding similar cases in the future.

Our work is motivated by collaborations with the CSU University Transportation Center [8]. This UTC is specifically focused on improving safety in work zones. Through the UTC, we worked with a local flagging company, Area Wide Protective (AWP), to define the problem space and to identify requirements (Section 2).

We have designed a complete sensornet system for monitoring work zones. Our system collects data in work zones, and presents them for two kinds of uses: First, we provide summary information of traffic activity around the work zone for post facto analysis for correlating near-crash instances with work zone design, and second, we publish traffic statistics to the internet. Where wireless internet connectivity is available in the work zone, our system publishes real-time statistics to MSR SensorMap [21].

We make the following contributions in this paper:

1. The design and prototype implementation of a sensornet system to monitor traffic in short-term work zones.
2. Software architecture (implemented in TinyOS/nesC) for collecting a variety of traffic statistics, such as flow, density, vehicle trajectories, etc.
3. Examples of real deployment experiences with temporary work zones.

The rest of the paper is organized as follows. We outline the design requirements that distinguish sensornet deployments in short-term work zones in Section 2. Following this, we describe our system architecture, and hardware and software design in Section 3. We present results from our evaluation and deployment experiences in Section 4. After discussing some of the lessons we learned during this research in Section 5 and related work in Section 6, we conclude in Section 7.

2 Design Requirements

Deployment Requirements. Short-term work zones present some design requirements that are distinct from typical sensornet deployments:

1. *Rapid deployment.* These work zones are only active for a few hours and the network must be ready in a few minutes.
2. *Inexpensive.* The cost of sensornet hardware must be kept to a minimum.
3. *No skilled maintenance.* The nodes in these systems must not require any skilled maintenance from sensornet experts.
4. *Self-organization.* While sensornet deployments in short-term work zones are not completely ad hoc, they are not guaranteed to be "highly-engineered": the placement of nodes in the network cannot be pre-determined.

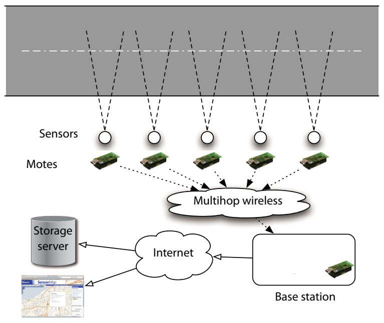


Fig. 1. Deployment architecture. Data collected from the work zone is uploaded to a server for archival and analysis and to SensorMap.

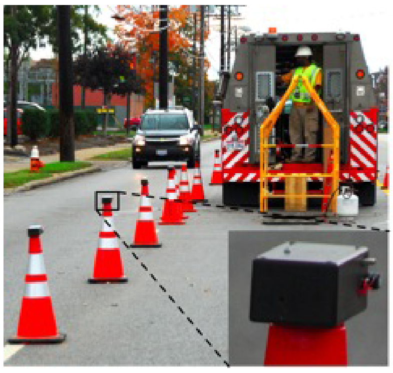


Fig. 2. Our sensornet system deployed in a work zone. The motes are mounted on safety cones, which are placed as per the MUTCD.

Data Requirements. Based on our discussions with the researchers at the CSU UTC, the most important kinds of data that needed to be collected were:

- Traffic statistics such as *flow* (vehicles per hour), *density* (average vehicles per mile), and *average speed* of vehicles traveling through the work zone.
- Trajectories of vehicles as they travel through the work zone. When cars deviate from a uniform straight line, there is potential for crash incidents since they may come close to construction equipment or workers.
- Aberrant behavior of vehicles. The design of a work zone is intended in such a way that vehicles will still be able to maintain uniform speed. Cases where vehicles suddenly brake, for example, may be indicators of unsafe situations.

3 System Architecture and Design

Given the design requirements for this problem, we wanted to come up with the simplest design of a sensornet that would still be able to provide the appropriate kinds of data required. In order to gather traffic statistics such as flow, density, and average speed, a simple array of proximity sensors can be used to count vehicles that move past the array. In order to compute vehicle trajectories, the proximity sensors would not be sufficient themselves, since the distance from the sensor to the vehicle obstruction will also be needed. Accordingly, we use an array of ranging sensors (Section 3.1).

Along the roadway of interest, an array of nodes with ranging sensors is deployed (Figure 1). Each sensor node is also capable of transmitting the sensed samples to a local base-station. The base-station is connected to a centralized server that is responsible for data archival and analysis. Most construction and utility trucks are equipped with a GPS receiver and a broadband internet connection, and our base-station can use this connection to access the Internet.

Sensor Placement. While the sensor node placement in the network is not highly-engineered, they are placed in a predictable manner. The nodes are placed along the side of the roadway being monitored such that the following assumptions are met:

- The entire width of the roadway falls inside the sensitivity region of the sensors.
- Separation between nodes in the network is uniform.

This deployment architecture, and the assumptions it makes, is quite well suited for the target application. For one, the sensing hardware can be integrated easily in the work zone: they can be mounted on the safety cones. Further, road construction personnel in work zones already have specific parameters that they need to meet in order to put together a safe work zone. There are guidelines on distance between cones, and placement of cones. The Manual of Uniform Traffic Control Devices (MUTCD, chapter 6) describes the rules of how to place traffic control devices (safety cones, in this case) in short-term work zones [9]. These guidelines and practices can be easily exploited in the design of our deployment. Figure 2 shows a photograph of our sensor nodes deployed in a live work zone run by our collaborators, Area Wide Protective. We did not modify the work zone, and the placement of cones in any way.

The black box (shown in inset in Figure 2) contains our sensor node hardware. The box itself it fastened to a plastic cup. This cup is placed on top of the safety cone. When mounted on the cone, the box is stable, while still being extremely simple to mount.

3.1 Hardware Design

Figure 3 shows the internals of the sensor node unit used in our sensornet deployments. Each node has the following elements, all of them off-the-shelf components:

1. *Processing and communication unit:* We use a TelosB mote [24] in the box. The mote's USB connector is exposed outside the box for programming and charging.
2. *Sensors:* We use an infrared ranging sensor to detect obstructions through holes drilled on the side of the box. There is also a magnetometer to classify obstructions.
3. *Battery source:* We use an Ultralife rechargeable lithium battery as the power source.

Detecting Vehicles. To detect vehicles, we use a ranging sensor that can not only detect an intrusion, but also provide the distance of the intrusion from the sensor. We use a Sharp infra-red ranging sensor (GP2Y0A700K0F) [26]. This sensor has a sensing range of 5.5 m, and provides an analog voltage signal (0.5–3V) based on the distance of the reflective obstruction. Figure 4 shows the output voltage profile that the sensor provides as a function of the distance of the obstruction. The sensor needs an input operating voltage of 4.5–5.5V. We use a Maxim MAX756 step-up converter to convert the voltage from the battery to the required input voltage for the sensor.

Battery Maintenance. The battery in the node is the only component that needs regular maintenance. In order to simplify maintenance, and to avoid replacing batteries in the box often, we use a rechargeable battery. Further, we connect a Telos charger board [23] to the TelosB mote, and connect the rechargeable Ultralife lithium battery to the board. Whenever the TelosB mote is plugged into a USB port, the battery is charged, and when the mote is not connected, the battery powers the mote.

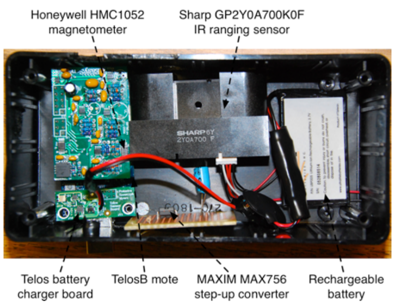

Honeywell HMC1052 magnetometer
Sharp GP2Y0A700K0F IR ranging sensor
Telos battery charger board
TelosB mote
MAXIM MAX756 step-up converter
Rechargeable battery

Fig. 3. Our sensor node includes a TelosB mote with a ranging sensor and a magnetometer

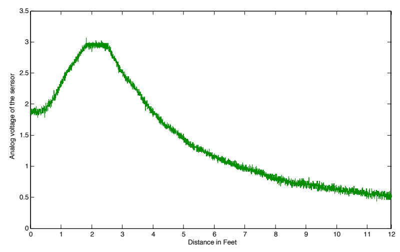


Fig. 4. Output voltage produced by the Sharp GP2Y0A700K0F infra-red ranging sensor

3.2 Software Services and Design

Collecting Sensor Data. Each node in the network periodically samples its two sensors to detect vehicles moving past the node. The collected samples are sent to the base-station for processing. A point to note here is that this network runs at a duty cycle of 100%. The goal of this system is to capture complete information about vehicle traffic, over a short period of time. Short-term work zones are only active for a few hours at a time. Between deployments, the batteries in the sensor nodes can be recharged.

Time Synchronization. The work zones that our sensornet targets are short-term work zones deployed in urban streets. The maximum speed on these roads is 35 mph. Given this speed, and the sampling duration of the ranging and magnetometer sensors, we sample our sensors at 10Hz. The typical distance between nodes in short-term work zones is about 10 feet. Based on this inter-node distance, a car moving at 35 mph will take about 195ms to travel from one node to the next[1]. We use the stabilizing clock synchronization protocol [12], which achieves an accuracy of 300μs.

Network Self-Organization. Our first attempt at network organization was to use a neighbor localization algorithm using RSSI between nodes based on [13]. In our setup, neighboring nodes are roughly 10 feet apart. So each node needs to identify its two nearest neighbors (the two nodes that are 10' on either side), and distinguish them from nodes that are further away. The key requirement, therefore, is that a node p should be able to distinguish between a node q that is 10' away from node r that is 20' away.

However, our own observations were not as consistent as [13]. In fact, we were not able to distinguish between RSSI readings at all between nodes 10 and 40 feet away (Figure 5). What we observed was consistent with [28]: RSSI is good indicator of link quality at some levels, but it is not a good indicator of distance (at least at the granularity we were interested in). In [1], the authors discuss using statistical methods or neural networks to estimate distance. We abandoned this approach since these algorithms made our system too complex, and opted for a more simple, centralized, approach to network organization: using the time-stamp information contained in the sensor messages to order the nodes at the base-station (described below).

[1] 35 mph is about 51 feet per second, so the time to travel 10 feet is 10/51 s $\approx$ 195 ms.

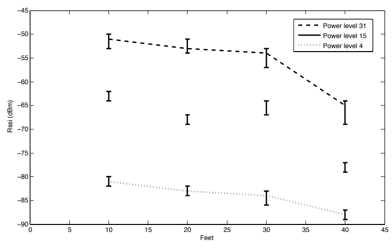


Fig. 5. RSSI at different distances

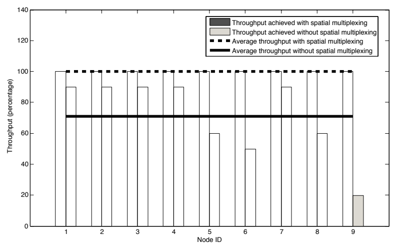


Fig. 6. Spatial reuse and goodput

We use a simple minimum spanning tree as the routing structure to transfer data from the network to the base-station. Once the routing tree is formed, the base-station disseminates two pieces of information to the network: (i) the depth of the routing tree, and (ii) the inter-node distance. The base-station is provided with the inter-node distance at the time of deployment. This is the only parameter that the system needs. We keep this a deployment-time parameter because the exact physical separation between the safety cones is only known at the time of commissioning the work zone.

Once the routing tree is formed, and the nodes are synchronized, they begin sampling their sensors to detect vehicle traffic. The samples are reported via multi-hop routing to the base-station, which can reconstruct vehicle paths using the time-stamp information available in the messages. Further, using the time-stamp information, the base-station can discover the topology of the network and the ordering of the nodes in the array: the time-stamps from different nodes tracking the same car will be in the order that the nodes are placed, since all nodes are synchronized. This simple sorting based on time-stamps, and using the target being tracked itself for localization, turned out to be more accurate than using other distributed localization schemes. We use the first 50 samples from each node as a training set for the base-station to converge on the topology of the network[2]. Once the training period is complete, the base-station disseminates the topology information to the network.

Sensor Sampling. The function of each node is to detect an obstruction that is crossing its "line of vision," classify it as a vehicle, and to estimate the distance (from the sensor) at which the vehicle crossed. The ranging sensor that we described earlier (Section 3.1) provides the distance information. For classifying the obstruction as a vehicle of interest, we use the magnetometer. The magnetometer sample is only used for classification and is not reported. Each data sample consists of three fields:

Timestamp (4bytes)	Distance (1byte)	Vehicle count (2bytes)

Every sensed sample is logged to the external flash on each node. This is done so that post facto analysis can recover data samples missed due to lost network packets. In addition, each node keeps a growing buffer of the recent samples that have not yet been uploaded to the base-station. These samples are uploaded to the base-station in batches.

Data Reporting. Each cycle of data acquisition needs to sample the sensors, and then report the sampled data to the base-station, if a vehicle is detected. If all the nodes were

[2] The training set is this big to remove errors caused by dropped messages, missed samples, etc.

transmitting their sensed data to the base-station at the end of every sensing cycle, the amount of wireless traffic in the network will be too high, leading to poor goodput. In fact, in our initial experiments, this is exactly what we observed: the yield of the network, even a small one with 9 nodes, was only around 73%. Instead, we use a delayed reporting scheme with the goal of improving goodput. The reporting scheme we use makes for spatial multiplexing similar to the Flush protocol [15], which is designed for bulk data transfer over large numbers of hops. Our networks are simpler in that the number of hops to the base-station is about 4–6. We used a simplification of the spatial reuse scheme by scheduling exactly one node to upload data in each slot. Using spatial reuse, we were able to increase the goodput of messages to nearly 100% (Figure 6).

We design our batch uploads such that a node p_1 can transfer all the data it has to upload to its parent p_2 in the routing tree within the duration of time that a vehicle will take to travel from p_1 to p_2. In this manner, if p_1 begins the transfer immediately upon seeing a vehicle, then the transfer can be completed before p_2 can see the same vehicle. The default TinyOS active message payload size used in the TelosB mote is 28 bytes [17]. In addition to the three fields above, each node will also need to include its node id (2 bytes) in each message. So each message can carry up to three vehicle samples (21 bytes), and the size of each message is 41 bytes including header and footer sizes. If the nodes in the network are placed d_{node} apart, the minimum time a vehicle takes to travel this distance is T_{node}, the time taken for a message to travel from sender to receiver is t_m, and h_{max} is the maximum hop count of any node in the network to the base station, then the size of the buffer on each node is at most $b = 3 \times \frac{T_{node}}{t_m \times h_{max}}$.

In most urban work zones, the safety cones (and consequently, the sensor nodes) are placed 10 feet apart (d_{node}), and the typical speed limit is 35 mph. So T_{node} is about 195ms. The message delay (t_m) is about 8 ms for the 41-byte message [4], and in most of out test networks, the height of the routing tree (h_{max}) is 3. So the size of the buffer on each node is 24: each node can cache 24 vehicles for each reporting cycle.

We employ a mutual exclusion scheme to schedule data transfers from each node. Only the node that has the mutex token transfers data, and the other nodes in the network are either idle, or are participating in multi-hop routing. The first node in the network assumes the token to begin with. When this node has accumulated b samples, it begins the transfer process. The message transfer process is started immediately upon completing a sample; this way, the sender node knows that its immediate neighbor in the array will not see the same vehicle for T_{node}, by which time all the data would have been transferred. After sending all the messages ($b/3$), it sends the mutex token to the next node in the array. The next node in the array now can begin its own data transfer process. This process continues until the last node has had a chance to upload its data. Notice that all nodes in the network can transfer their cached data in the time it takes for a single car to move through the network. Once the last node in the network has transferred all of its data in that round of transfers, the base-station disseminates a completion signal. This completion signal serves to hand the mutex token back to the first node in the network, and the reporting cycle repeats approximately every b vehicles.

Computing Vehicle Trajectories. The basic idea behind our trajectory tracking system is quite simple: Whenever a vehicle crosses the sensing region of a sensor, the mote takes a sample and sends a “sensor-to-target distance” measurement to the base-station. This message is packaged along with the node’s ID, and its local timestamp. For now, let us consider the simplest formulation of this problem: that there is only one vehicle moving through the array. We will discuss multiple targets later.

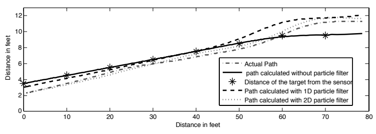
The base-station learns the network topology during the training period, and can locate a node with ID k to a particular (x_k, y_k) coordinate location. This coordinate location, in addition to the target distance, can be used to compute the target’s coordinate location at time t_k: $(x^{t_k}_{target}, y^{t_k}_{target})$. With sensor readings from all the nodes in the array, the base-station can assemble an ordered list of points through which the target traveled. A simple curve passing through these points will give us an approximation of the actual path the vehicle took. However, the sensor, based on our calibration, has an error margin of about a foot. This is nearly 7% of the entire sensing range!

To improve the accuracy of the trajectory mapping algorithm, we implemented a *particle filter* [7] algorithm based on the one in [27]. At the base-station, the particle filter generates a set of random points for each sensor sample. Based on a cost function (described below), the particle filter then prioritizes these points. The point from each set with the least cost function is picked as the candidate trajectory. In [27], the authors use a particle filter in order to detect multiple targets in a 1-dimensional space using binary proximity sensors. Our space is a 2-dimensional space, and we modified the setting accordingly. In our 2-dimensional space, the sensors are arranged along the x-axis, and we consider that the sensor’s range is a straight line along the y-axis. We use a combination of two cost functions, one for each of x and y dimensions.

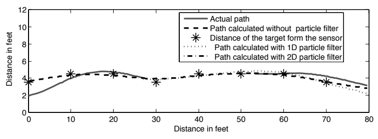
In the x dimension, we use an approach that uses the speed information of the target to bias the cost function. Based on the speed of the target calculated by pairs of sensors, the particle filter can find the most probable location of the target in the particle space. The trajectory computation is done at the base-station post facto, and the speed information is already available by then. In the y dimension, the cost function serves to eliminate changes in the trajectory of the target that are unrealistic. Most passenger cars are about 12’ to 16’ long. Given that the distance between our sensors are about a car-length or less, the amount of variance in the sensor-to-target distance reported by the sensor is limited. The cost function we use in the y dimension, therefore, is weighted to limit such unrealistic variations. At the same time, we do not want the filtering to miss actual variances in target trajectories (which is the whole point of this exercise).

Detecting Aberrant Behavior. Sudden changes in speed of vehicles typically indicate potentially unsafe physical situations on the roadway. By calculating speed between every pair of sensor nodes in the array, we can get the speed of the moving vehicle in different regions of the work zone. Normally, one would observe a uniform speed, or a gradual increase or decrease of speed. Sudden fluctuations (*e.g.,* 10% change within 20 feet) are triggers to flag a vehicle as moving in an aberrant fashion. The number of such instances are recorded, along with where in the work zone they occurred. By examining this data, deductions can be made about potential safety hazards in work zone design.

Publishing Data for Wide Access. On a typical day, there are tens, even hundreds of short-term work zones that are active. Our partner, AWP, alone deploys a number



(a) Path 1



(b) Path 2

Fig. 7. Comparing the actual path of a target across the sensor array, and the trajectory we computed. We drove our target along two different paths, and tested the three versions of trajectory mapping to approximate the target's path.

of active work zones in the Cleveland area. One of the biggest problems with short-term work zones is that there is typically no record of its existence. In fact, except for motorists that are driving along the street on which the work zone is commissioned, no one even knows about it. While traffic information on major highways in metro areas is already available in mapping services such as Google Maps [10] and Microsoft Live Maps [18], traffic delays caused by short-term work zones are not reported. Our base-station uploads synthesized traffic data to the internet (Micosoft's SensorMap [21]).

4 Evaluation and Results

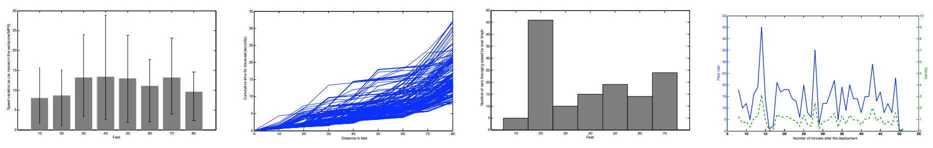
4.1 Estimating Vehicle Trajectories

We tested our trajectory mapping algorithms in a testbed deployed in a parking lot with eight nodes. The nodes were placed ten feet apart from each other, in a straight line. One of the motes acted as the root of the collection routing structure, and communicated with a PC acting as the base-station. We drove a car in a pre-determined path as our target moving through the sensor array. Each sensor took 10 samples/sec. This sampling rate was sufficient to capture the target moving through the array, based upon the speed of the target moving across the array, and inter-node distance.

Figure 7 shows the results of our experiments with two paths. In the case of each of the paths, four curves are shown. One of these is the actual path traveled by the target vehicle. The first calculated curve simply takes the sensor readings, directly. These readings, based on our sensor calibration tests, may be off by up to one foot from the actual path. The second calculated curve uses 1-d particle filter (along the y-axis) to better approximate the reading, and to compensate for sensor calibration error margins. The final calculated curve is the curve calculated using the 2-d particle filter. As one can see from all the three different paths we tested with, the accuracy of the computed path becomes better as we move from plain sensor calibration, to 1-d particle filtering, and finally to 2-d particle filtering.

4.2 Deployment Experiences

We deployed our sensornet system on work zones commissioned by Area Wire Protective (AWP), a flagging company in Northeast Ohio. The company provides road work zone services to a number of utility companies in the area. When a utility company



(a) Average vehicle speeds in different segments

(b) Cumulative time of traversal for vehicles driving across the work zone

(c) Number of cars that changed speed by over 5 mph in adjacent 10 ft segments

(d) Flow and density in different work zone segments

Fig. 8. Some sample statistics we are able to collect using our sensornet

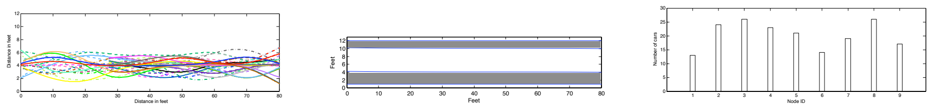
(gas, electric, cable) has to perform maintenance work that may cause traffic restrictions, AWP sets up a work zone for them to ensure safe operation.

In this section, we report data we collected from one of these work zones in the Greater Cleveland area[3]. The location of the work zone was on Lorain Road near the intersection with Clague Road in North Olmsted. This is a pretty busy road, and in one hour during our deployment, we observed 614 cars pass through the work zone. This work zone was about one hundred feet long, and occupied one lane of the street. The work area was in one of the drive lanes, and was about 20 feet long. The street had two drive lanes in either direction, and a turn lane in the middle. The work zone guided the traffic to merge from two lanes into one. We deployed our sensors to monitor traffic in the lane that carried the merging traffic. We videotaped the traffic during the deployment to compare with the data produced by the network to establish ground truth using traditional methods of traffic analysis.

Average speed of vehicles. The speed limit on Lorain Road is 35 mph, and there was no reduction in speed limit caused by the work zone. There was a traffic light about 500 feet downstream from our work zone, and this caused some slowdowns and some stopped traffic as well. The average speed of vehicles driving through our work zone was about 12 mph. Further, we measured speeds between every pair of nodes, *i.e.,* average speed in every 10-foot segment in the work zone. These speeds are shown in Figure 8a. Notice how the average speed of vehicles is lower immediately upon entry into the work zone, and just before exiting the work zone. Near the middle of the work zone, motorists generally tend to be “more confident,” and hence tend to speed up a little. In spite of the speed limit being 35 mph, we didn’t actually observe any vehicles traveling as fast. This was mostly because of the density of traffic, which was “bumper-to-bumper” for most of the time the work zone was active. As another measure of how fast vehicles are moving through the work zone, we show cumulative time-location plots of vehicles in Figure 8b. Looking at this figure, we can see that most cars spend about 10–20 seconds in the work zone, while a small number of them spend longer.

Changes in speed. Sudden changes in vehicle speeds is another point of interest for work zone designers. If a number of vehicles suddenly changed speed particular spot in

[3] The data collected from two other test deployments are similar in kind. The complete collection of datasets is available at `http://selab.csuohio.edu/dsnrg`



(a) The trajectories of 50 cars out of our total set of 614. Note that this is just the side of the cars facing the sensors.

(b) The white region in the middle denotes the average trajectory. The shaded regions on denote the extent of deviation from the average.

(c) Number of cars that went off the average trajectory as detected by each node in the work zone

Fig. 9. Trajectories of vehicles in the work zone

the work zone, that spot merits some special consideration. Figure 8c shows the number of cars that changed speed by over 5 mph in adjacent 10 ft segments. See the correlation between this graph, and the graph in Figure 8a: a number of cars speed up in the second segment, resulting in a higher average speed in the middle of the work zone. Near the end of the work zone, a number of cars reduce speed just before exit.

Rate of flow and density of traffic. Figure 8d shows the rate of flow of traffic, and traffic density, during the hour of data capture. As we can see here, for most of the time, the work zone had a fair number of cars driving through it. There are very short intervals of time when the flow rate was less than 5 cars per minute. This is a good way for us to validate our sensor sampling rate. Even in dense traffic, our sensornet is able to produce good data. As we said earlier, we videotaped the traffic during this time, and compared it with the data collected from the sensornet. We found the data to be very well correlated with the video data.

Vehicle trajectories. Using the trajectory mapping scheme described in Section 4.1, we calculate trajectories of the vehicles driving past our sensor array. During our deployment case study, we observed a majority of vehicles maintaining a steady path through the work zone. The average trajectory was about 4 feet from the side of the lane (Figure 9). The width of the lane is 12 feet, and the average car is about 6 feet wide. Given this, if cars were driving perfectly in the middle of a lane, then they would be 3 feet from either edge of the lane. The tendency of most drivers, when they see safety cones or other construction equipment, is to tend away from them, and favor driving closer to the opposite edge of the lane. This anecdotal tendency is confirmed in our case study instance, where the average trajectory is a foot further than the centerline of traffic in the lane away from the safety cones.

However, a number of cars did veer off the average trajectory, and some came too close to the safety cones, and some others were driving too close to the opposite curb. Figure 9c shows the number of cars that veered too close to the safety cones measured at each of the sensor nodes. These instances are of interest to work zone designers: if there were an inordinate number of vehicles leaving the preferred trajectory at a single spot, that may indicate a potential unsafe situation. In our case, there is no such unusual observation, indicating that the traffic in this work zone was mostly compliant.

5 Discussion

Throughout this research, during design and development, we constantly worked along with transportation engineers from the CSU UTC to make sure that we meet our design guidelines. Comparing with the list in Section 2, our system meets them quite nicely:

1. The nodes are programmed to collect traffic statistics, and the base-station is programmed to synthesize this data for publication on the internet. Deployment of the network simply entails mounting the nodes onto safety cones and turning them on.
2. Our prototype node is built from off-the-shelf parts, and as such, the cost of all parts in the node add up to about \$220. We expect to cut this cost in about half with mass-fabrication. In comparison, most sensor equipment deployed in long-term work zones run thousands of dollars per node.
3. The only regular maintenance that is needed for our sensor nodes is to keep the battery charged. We have conveniently exposed the USB connector of the TelosB mote for this purpose. Simply plugging in the mote will charge up the battery.
4. Our network does not expect to be deployed in a pre-determined fashion. Instead, we built our self-organization logic around the practices that the work zones follow. Accordingly, we know that the nodes will be placed at uniform distances apart from each other, and this distance is provided to the network as a parameter.

As an alternative to the infrared sensor, we are currently looking into other ranging sensors that are similarly inexpensive and easy to use. As of this writing, we are experimenting with the SRF02 ultrasonic range finder [6]. This sensor has a similar range as our Sharp infrared range finder (6 m). The sensor can connect to the TelosB mote through the I^2C interface, and directly provides a distance reading (in cm) based the obstruction in front of the sensor. The sampling time of this sensor, however, is twice that of the IR sensor, which may cause timing issues with respect to capturing traffic: this sensor may miss some vehicles because of the reduced sampling rate.

While we were working on writing this paper, we came across advice for successful sensornet deployments [2]. We were quite pleasantly surprised that we had already followed a number of the good practices [2] listed. For example, from the beginning, we have worked with domain specialists from transportation engineering to define the problem, and to expose the solution spaces; we have always trusted experimentation with real hardware as opposed to simulation (we used a smaller version of the IR sensor for lab tests with toy cars early on); our protocols are as simple as they can be, making system behavior very predictable.

6 Related Work

In the past, most traffic monitoring systems were built with high-cost equipment such as measuring poles, inductive loops, etc. [25]. Moreover, these deployments are *invasive*, requiring activities such as road digging which require high labor. In our system sensor nodes can be deployed with minimal engineering efforts (at most placing them on either side of road or along a straight line on one side of the road). And compared to deployments involving inductive loops, sensornets come at a substantially lower cost.

Indeed others have used sensornets in the traffic monitoring context. In [5] and [16] authors have deployed sensors in the intersections of freeways and parking lots. [11] describes wireless magnetic sensors that can be used for traffic classification and surveillance. The sensors are designed to identify vehicles, speed of the vehicles, conditions of the road, density of the traffic etc. But the problem we deal with is, apart from monitoring the traffic, we are interested in the trajectories of the vehicles especially in work zones so that the traffic authorities can now learn about "near-crashes" which are impossible to find. There are also other mechanisms which directly deal with the driver rather than the vehicle [20]. These systems simulate traffic and may be subject to errors.

In [29], Yoon et al. show how to estimate traffic on streets using GPS traces. In their system, cars are equipped with GPS receivers, and the traces of these GPS receivers is used to analyze traffic patterns. The Nericell [19] system is similar in that they use sensors in moving vehicles. As opposed to [29] and [14], however, their focus is on using sensors that people carry with them anyway. Their work is focused on using smart cell phones (which have a number of sensors such as GPS, microphone, accelerometer, etc.) to derive vehicle traces. By using this heterogeneous sample of traces, they can identify potholes on roads, distinguish traffic stopped at a red light from traffic stopped in a jam, etc. All these systems are complementary to our work, since they involve embedding sensors in moving vehicles.

The California Department of Transportation [3] maintains a website with live feeds from a number of sensors across the state of California and a wealth of information in the form of studies and reports focused on monitoring traffic. The Ohio Department of Transportation (ODOT) maintains a similarly rich web-accessible system called Buckeye Traffic [22], and provides current information about road closures and restrictions on major highways because of construction projects, and identifies road activity from a variety of permanent sensors all over Ohio. However, no short-term work zones are captured; it is not feasible to have permanent sensors deployed in every street.

7 Conclusion

Construction work zones on roadways are hazardous areas. Motorists driving through a roadway under construction may end up facing unexpected scenarios. Long-term work zones on major highways may present such unfamiliarity in the beginning, but once a motorist has driven on the modified road a few times, she can get used to the changes (which will last a few weeks, if not months). By contrast, *short-term* work zones —the kind that we see in our local city streets for utility work— are only active for a few hours at a time. This transient nature leaves them untraceable for the most part. In fact, there is very little empirical data available about traffic in short-term work zones.

We have presented the design and prototype implementation of a sensornet system that is specifically targeted at collecting data about traffic in and around short-term work zones. Our system is rapidly deployable, easily maintainable, and is capable of capturing a variety of different statistics about vehicle traffic in work zones. The data collected can be used by transportation engineers to consider design parameters for future work zone configurations. We have tested our systems in live work zones in the Cleveland area, and are now in the process of working on expanding to wide deployment.

References

1. Awad, A., Frunzke, T., Dressler, F.: Adaptive distance estimation and localization in wsn using rssi measures. In: DSD 2007, pp. 471–478. IEEE, Los Alamitos (2007)
2. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M.: The hitchhiker's guide to successful wireless sensor network deployments. In: SenSys 2008 (November 2008)
3. California Center for Innovative Transportation. Traffic surveillance, `http://www.calccit.org/itsdecision/serv_and_tech/Traffic_Surveillance/surveillance_overview.html`
4. Chebrolu, K., Raman, B., Mishra, N., Valiveti, P.K., Kumar, R.: Brimon: a sensor network system for railway bridge monitoring. In: MobiSys 2008, pp. 2–14. ACM, New York (2008)
5. Cheung, S.-Y., Varaiya, P.: Traffic surveillance by wireless sensor networks: Final report. Technical Report UCB-ITS-PRR-2007-4, University of California, Berkeley (2007)
6. Devantech. SRF02 (2008), `http://www.robot-electronics.co.uk/htm/srf02techI2C.htm`
7. Doucet, A., Godsill, S., Andrieu, C.: On sequential monte carlo sampling methods for bayesian filtering. Statistics and Computing 10(3), 197–208 (2000)
8. Duffy, S.F.: Csu transportation center (2007), `http://www.csuohio.edu/utc`
9. Federal Highway Administration. Manual on Uniform Traffic Control Devices. U.S. Dept of Transportation, Washington D.C., 2003 edition with revisions 1 and 2 edition (December 2007)
10. Google. Google maps (2008), `http://maps.google.com`
11. Haoui, A., Kavaler, R., Varaiya, P.: Wireless magnetic sensors for traffic surveillance. Transportation Research Part C: Emerging Technologies 16(3), 294–306 (2008)
12. Herman, T., Zhang, C.: Stabilizing clock synchronization for wireless sensor networks. In: Datta, A.K., Gradinariu, M. (eds.) SSS 2006. LNCS, vol. 4280, pp. 335–349. Springer, Heidelberg (2006)
13. Holland, M.M., Aures, R.G., Heinzelman, W.B.: Experimental investigation of radio performance in wireless sensor networks. In: SECON 2006 (2006)
14. Hull, B., et al.: Cartel: a distributed mobile sensor computing system. In: SenSys 2006, pp. 125–138. ACM Press, New York (2006)
15. Kim, S., et al.: Flush: a reliable bulk transport protocol for multihop wireless networks. In: SenSys 2007, pp. 351–365. ACM Press, New York (2007)
16. Knaian, A.: A wireless sensor network for smart roadbeds and intelligent transportation systems. Technical report, MIT EECS and the MIT Media Laboratory (May 2000)
17. Levis, P.: Tinyos tep 111 (2008), `http://www.tinyos.net/tinyos-2.1.0/doc/html/tep111.html`
18. Microsoft. Microsoft live maps (2008), `http://maps.live.com`
19. Mohan, P., Padmanabhan, V., Ramjee, R.: Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In: SenSys 2008 (November 2008)
20. Nadeem, T., Dashtinezhad, S., Liao, C.: Trafficview: A scalable traffic monitoring system. In: 2004 IEEE International Conference on Mobile Data Management, pp. 13–26 (2004)
21. Nath, S., Liu, J., Miller, J., Zhao, F., Santanche, A.: Sensormap: a web site for sensors worldwide. In: SenSys 2006, pp. 373–374. ACM Press, New York (2006)
22. Ohio Dept of Transportation. Buckeye traffic (2008), `http://www.buckeye-traffic.org`
23. Polastre, J.: Telos charger board (2005), `http://www.tinyos.net/hardware/telos/sensorboards/UCB_TSB+TCB_schematic-2005-07-31.pdf`

24. Polastre, J., Szewczyk, R., Culler, D.: Telos: enabling ultra-low power wireless research. In: IPSN '05, Piscataway, NJ, USA, IEEE Press, Los Alamitos (2005)
25. Santel, G.: Traffic flow and accident occurrence in construction zones on freeways. In: 6th Swiss Transport Research Conference (March 2006)
26. Sharp. GP2Y0A700K0F, `http://www.acroname.com/robotics/parts/R302-GP2Y0A700K0F.pdf`
27. Singh, J., Madhow, U., Kumar, R., Suri, S., Cagley, R.: Tracking multiple targets using binary proximity sensors. In: IPSN 2007, pp. 529–538. ACM Press, New York (2007)
28. Srinivasan, K., Levis, P.: Rssi is under-appreciated. In: EmNets (2006)
29. Yoon, J., Noble, B., Liu, M.: Surface street traffic estimation. In: MobiSys 2007, pp. 220–232. ACM Press, New York (2007)