Utilizing Correlations to Compress Time-Series in Traffic Monitoring Sensor Networks

A. Guitton, A. Skordylis and N. Trigoni
Birkbeck College, University of London

Abstract— **Wireless sensor networks present significant opportunities for fine-grained and continuous monitoring of road traffic, enabling careful city planning, automated road maintenance and accident detection. Users are typically willing to tolerate a small error in car-flow data, in order to reduce the cost of data propagation from the sensor nodes to the gateway nodes, to which users are connected. In this paper, we first examine the relative performance of Fourier- and Wavelet-based algorithms for compressing traffic data locally at the sensor nodes. Using real traffic information from the city of Cambridge (UK), we then demonstrate that car-flow data collected across geographically dispersed sensor nodes exhibit strong spatial and temporal correlations. We then combine lossy Fourier-compression with correlation-based compression to achieve further communication savings within a user-specified error threshold. Two distinct algorithms are proposed, one exploiting same-node temporal correlations and the other spatial correlations across nodes. For a tolerated error of 5-15 cars / 5 min, it is shown that exploitation of temporal correlations yields 14-30% savings relative to Fourier compression alone, whilst use of spatial correlations results in 10-35% savings. Both algorithms are fully distributed and easy to implement in resource-constrained sensor and relay nodes.**

I. Introduction

The annual cost of road congestion in the UK is estimated at £20bn. In urban areas, transport is the major source of carbon dioxide emissions, and traffic monitoring can give useful insights on how to extend the road network or apply road usage restrictions to maintain pollution within acceptable limits. The availability of traffic information is also paramount to preventing or handling traffic jams observed after accidents, concerts and soccer games.

It is evident that urban areas would considerably benefit from a sensor network infrastructure able to detect vehicle flow, speed and occupancy at high spatial and temporal resolutions. In this paper, we investigate how to use spatio-temporal correlations in traffic data to reduce the running costs of the monitoring infrastructure given user-defined accuracy requirements. We provide a framework for disseminating traffic data through a sensor network in an energy-efficient manner. Our specific contributions are as follows:

- We compare the communication savings and computation costs incurred by a Fourier-based and a Wavelet-based compression technique, which we run locally at sensor nodes with limited storage and processing capabilities. Our results use real traffic flow data and run on a real sensor platform. We provide quantitative results to show the compression rates of typical traffic time series given user-defined requirements for data accuracy.
- We study the extent of spatial and temporal correlations in real traffic data, and show how they differ from networks monitoring physical processes, like temperature fields. We propose distributed algorithms to identify spatio-temporal correlations, and exploit them to further reduce the cost of data propagation from the sensors to the gateway nodes. This work focuses on periodic traffic updates sent at the end of each day. Real-time traffic data propagation will be investigated in future work.

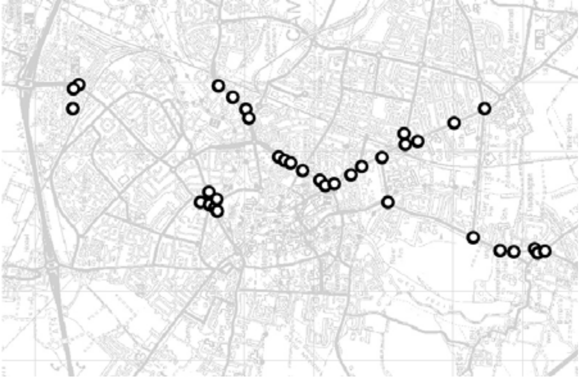

Fig. 1. The Cambridge traffic sensor deployment. Circles represent a sensor site monitoring one or more lanes on the road.

The remainder of the paper is organized as follows: In Section II we present the assumptions of our model, and provide a brief description of the traffic application scenario, including the existing Cambridge testbed (Figure 1) and the spatio-temporal properties of a real traffic dataset. Section III presents a class of correlation-aware data dissemination algorithms and compares their performance in terms of communication cost. We discuss related work in Section IV and present our conclusions in Section V.

II. Application scenario

In this section, we establish the main assumptions of our model: we describe our hybrid sensor network architecture and point out useful properties of urban traffic data.

Architecture: We assume a hybrid network architecture that includes three types of nodes: sensor nodes, gateway nodes and relay nodes.

Sensor nodes are equipped with a variety of sensor devices and they are targeted at monitoring road traffic in an urban scenario. For example, Figure 1 shows a real deployment of 112 sensor nodes in the city of Cambridge (UK), which

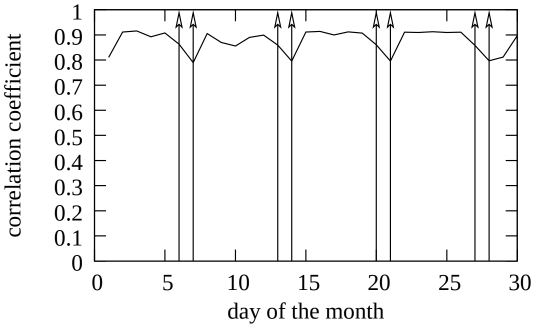

Fig. 2. Correlation coefficient between the time-series of 31st May and that of every other day in May.

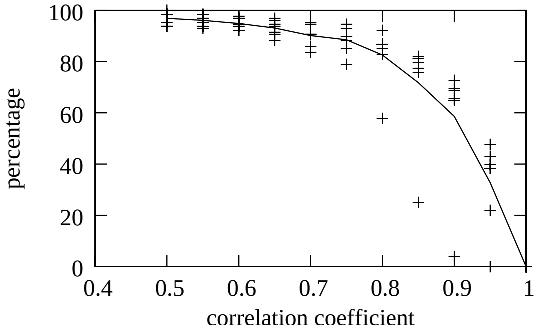

Fig. 3. Percentage of nodes with a given correlation coefficient between today's time-series and the time-series of a previous day within the last week.

span an area of roughly 3.5km×8km. Observe that most sensors are clustered around intersections, and different clusters are usually distant from each other. Sensor nodes are set up to measure traffic flow (cars/time) and street occupancy (cars/area) every 5 minutes. In this paper we focus on the readings concerning the flow of cars (i.e., number of cars observed/5min). Although the current sensor deployment involves wired sensors, our vision is to equip sensor nodes with radios and enable them to communicate their readings in a wireless multi-hop manner. In order to keep the installation cost low, we assume that sensor nodes typically have limited communication, storage and processing capabilities.

Gateway nodes collect the readings from the sensor nodes. They have a fixed power source, unlimited bandwidth, storage and processing capabilities, and ultimately route data towards a central server where it is further processed and stored.

Relay nodes are deployed to ensure that every sensor node remains connected to at least one gateway node. Relay nodes are required in our architecture because sensor nodes are sparsely deployed and some of them may not be able to establish a multi-hop wireless path to reach a gateway node. Relay nodes are less expensive than gateway nodes, but they are typically battery-powered and have limited communication, computation and storage capabilities. Unlike sensor nodes, they have no sensing capabilities, and they are only used for routing and processing purposes.

Spatio-temporal properties of urban traffic data: We now study the properties of a typical urban traffic dataset, focusing on temporal and spatial correlations in car flow data. We divide flow measurements to derive a time series per day per sensor. For example the tuple $SID, d, [f_1, f_2, \ldots,]$, denotes that sensor SID reported on date d readings f_1 in the first five minutes, f_2 in the next five minutes, etc. To better understand the traffic dataset, we measured i) the correlation between two time series of the same node on different dates; and ii) the correlation between two time series of different nodes on the same date. In both cases, we used the Pearson correlation coefficient between time series $F = [f_1, f_2, \ldots,]$ and $F' = [f'_1, f'_2, \ldots,]$:

$$r(F, F') = \sqrt{cov(F, F')^2/(var(F) \times var(F'))}$$

Temporal Correlations: We observed very strong *temporal correlations* in the time-series of a sensor node. For example, Figure 2 shows that a very high correlation coefficient ($cc \geq 0.9$) is observed between the time series of a node on Wednesday, 31st of May 2006, and any other day in May, except for weekends and May 1st, which is a bank holiday in the UK. Figure 3 summarizes the extent of temporal correlations in the traffic dataset. It shows the percentage of nodes that exhibit a given correlation coefficient between today's time-series and the time-series of a previous day within the last week. For more than 80% of the sensors, daily readings are highly correlated ($cc \geq 0.8$) with those on one of the previous seven days. These results show the strength of temporal correlations in traffic data, and reveal a unique opportunity for exploiting these correlations to achieve communication savings.

Spatial Correlations: The next question that arises is whether we can equally rely on spatial correlations between the time-series of different sensors, and whether the strength of spatial correlations depends on the physical distance between them. In order to determine whether two time series $f(M)$ and $f(N)$ generated by sensor nodes M and N respectively are correlated, we allow a time-shift between the time series. For example if sensors M and N are located along the same road and if the traffic flows from M to N, we expect that $f(M)[t] = f(N)[t + dt]$. Figure 4 shows that the correlation coefficient between two nodes does not depend on their distance. Unlike other applications, such as temperature monitoring systems, many pairs of remotely placed nodes were found to be highly correlated, whereas many pairs of nearby nodes exhibited little correlation. This can be explained by the fact that cars tend to remain in main roads, which traverse our deployment map from end to end.

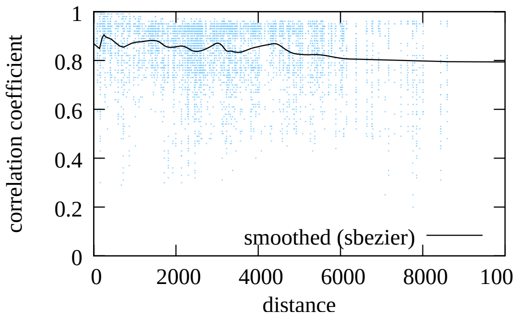

Fig. 4. Spatial correlation between two nodes as a function of their distance.

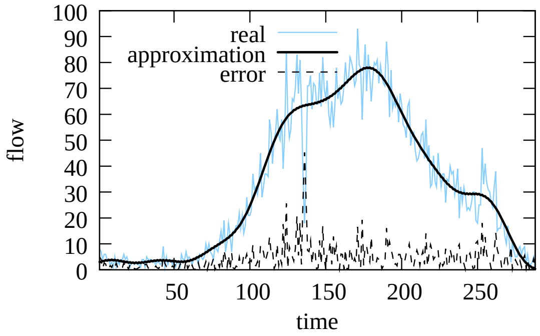

Fig. 5. The Fourier approximation of car-flow readings monitored by a sensor node every 5 minutes over a 24-hour period.

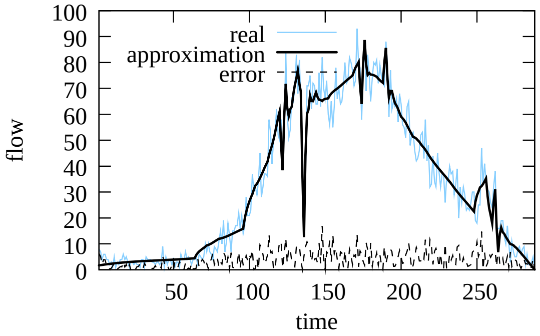

Fig. 6. The Daubechies-4 wavelet approximation of car-flow readings monitored by a sensor node every 5 minutes over a 24-hour period.

III. Correlation-Aware Dissemination Algorithms

In this section, we present a class of energy-efficient algorithms for sending periodic traffic updates from the sensor nodes to the gateway nodes. Gateway nodes are placed to minimize the sum of hop counts from sensor nodes to the closest gateway. All algorithms discussed in this section use tree-based routing: Each sensor node selects to forward its data hop-by-hop to the closest gateway node, i.e. the one accessible using the least number of hops. It forwards its data to its *parent*, which is the sensor or relay node on the min-hop path to the gateway node. Trees that connect sensor and relay nodes to gateway nodes through min-hop paths are generated during an initial network configuration phase using a simple flooding protocol, and they are maintained throughout the network's lifetime. Assuming that routes are set-up and continuously maintained, we study the problem of reducing the cost of data propagation by means of in-network lossy compression. In particular, we investigate in-network reduction of i) a single time series, ii) multiple time series generated by a single node, and iii) multiple time series generated by different nodes.

A. Reducing a single time series

Each node locally produces a large time series of traffic data per period, which, if propagated in an uncompressed form, would require a substantial amount of battery power spent on radio communication. Given the user's tolerance for a small error in the reported data, we examine two prominent signal compression techniques, one based on the Fast Fourier Transform (FFT) and the other on the Wavelet Transform (WT), and evaluate their efficiency in reducing the cost of radio communication.

We first examine the computation cost of Fourier and Wavelet transforms if they were to run locally on resource-constrained sensor nodes. We implemented both techniques in NesC for TinyOS, and measured their computation time on the Tmote Sky [1] platform, while varying the length of the input time-series. Computation time was measured to be linear in the length of the input time-series for all algorithms. Although the complexity of FFT is higher than that of WT ($O(n \log n)$ for FFT vs. $O(n)$ for WT), FFT computes a time series approximation twice as fast as WT using Daubechies-4 wavelet functions, which in turn is twice as fast as WT using Daubechies-8 wavelet functions. For instance, in order to process the time series of a whole day (24 hours) on a mote, FFT requires 4.6 seconds, whereas WT Daubechies-4 and WT Daubechies-8 require 10.5 seconds and 20.5 seconds respectively. This is attributed to the fact that WT uses mainly float operations, which are much slower on our platform than operations on integers.

Although FFT is a faster algorithm, it yields a higher maximum absolute error than wavelets as shown in Figures 5 and 6. The comparison is based on using the same number of bytes to represent the compressed time series. In the FFT case, the first n coefficients are used to approximate the time series, which require $n * sizeof(int)$ bytes. The first FFT coefficients correspond to the lower frequencies of the signal which account for the major signal variations. In the WT case, we retain the best m coefficients so that $m * sizeof(waveletCoef) = n * sizeof(fourierCoef)$.

FFT is an attractive choice for two reasons. Firstly, it has a lower computation cost than WT, and therefore is easier to compute in resource-constrained nodes. Secondly, it allows us to use the Fourier coefficients, instead of the raw time-series data, to speed up the evaluation of the correlation coefficient between two time series. Even though the maximum absolute error exhibited by FFT is significantly larger than that of WT, this error typically persists for less than 5 minutes, which is considered acceptable for traffic applications. The benefits of the FFT algorithm were therefore deemed to outweigh its limitations in this particular context, which led us to select it for in-network compression. However, it must be noted that other signal compression techniques could be used instead in scenarios where processing power is not a limited resource.

Fourier Compression (FC): The strawman algorithm that uses FFT for in-network compression does not exploit correlations between different time series. It propagates the Fourier coefficients that constitute the compressed version of a node's time series to the closest gateway through the shortest path. Say that nodes are requested to send updated traffic information at the end of each day with a certain error threshold ϵ. Then each node identifies the least number of Fourier coefficients k that can be used to reconstruct the time series with a maximum absolute error less than ϵ. It sends them to the closest gateway without performing any further reduction in the way, and temporarily stores them in local memory for

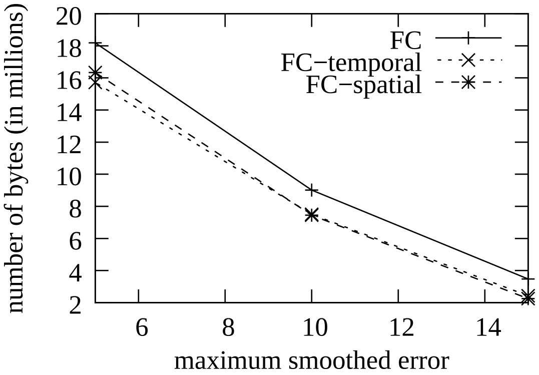

Fig. 7. Comparison of the total communication cost incurred by the three proposed algorithms using a single gateway.

fault-tolerance reasons. The FC algorithm compresses a single time series at the node where it is generated, without exploiting its similarity with previous time series generated in the same or different nodes.

B. Reducing multiple time series

In this section, we examine the use of spatio-temporal correlations to reduce multiple time series generated by the same or different sensor nodes.

Fourier Compression and Temporal Correlations (FC-Temporal): This algorithm extends the FC algorithm, in that it exploits temporal correlations to reduce the cost of time series propagation. Each sensor node running FC-Temporal tries to compress today's time series locally, by expressing it as a linear function of a previous day's time series.

More specifically, on the first day, each node computes k Fourier coefficients that can be used to reconstruct the original time series within the input error threshold ϵ, as in the case of the FC algorithm. On any of the following days, say day cur, the node iterates through the previous approximated time series $\{\hat{f}_{cur-w}, \ldots, \hat{f}_{cur-1}\}$, which are stored locally, and computes the correlation coefficient between them and the current time series $cc(f_{cur}, \hat{f}_d)$, $\forall d \in [cur-w, cur-1]$. As soon as it identifies a strongly correlated time series, say of day pr, it evaluates the regression parameters of the linear function that approximates cur's time series based on pr's readings: $\hat{f}_{cur}[j] = r_1 + r_2 \cdot \hat{f}_{pr}[j]$, where j ranges over all readings of a day's time series. If the maximum absolute error between the approximated and the original time series on day cur does not exceed the user threshold ϵ, the node sends to the gateway only the regression parameters of the linear fit: If $\max_j \left|\hat{f}_{cur}[j] - f_{cur}[j]\right| < \epsilon$, send (pr, r_1, r_2) to the gateway. When the gateway receives such a triplet, it retrieves $\hat{f}_{pr}$ from its cache, and estimates the current day's time series as $r_1 + r_2 \cdot \hat{f}_{pr}$ with adequate accuracy. If no linear correlation with a previous day is detected (within a window of w days), the sensor node approximates and forwards today's time series independently of previous days as in FC.

The FC-temporal algorithm requires that each node stores locally the approximate time series detected in the previous w days. This is realistic since, for traffic data, very strong correlations occur by setting $w = 7$. This algorithm is similar to FC, in that in-network computation occurs where data is first generated, and the algorithm does not try to merge data generated by different sensors to achieve further communication savings.

Fourier Compression and Spatial Correlations (FC-Spatial): The next step is to exploit spatial correlations only, i.e. correlations between the time series of different sensor nodes on the same day. We propose a fully-distributed algorithm, named FC-Spatial, which operates similarly on both sensor and relay nodes. Each intermediate node in the communication tree (sensor or relay node) receives approximated time series from its descendant nodes, and tries to further reduce them by exploiting their correlations. The only difference between sensor and relay nodes is that sensor nodes *receive* an additional time series from their local sensor device, which they compress using FFT, as discussed in FC.

Let an intermediate node I receive an approximate time series $\hat{f}(N)$ of today's traffic monitored by node N. This information is sent from N to I in the form of Fourier coefficients annotated with the Fourier compression error $\epsilon(N)$. Node I searches its local memory for correlated time series and, depending on the search result, it takes one of the following steps:

Step A: Suppose that the local cache includes another time-series $\hat{f}(M)$ of today's traffic monitored by node M, such that $\hat{f}(N) \approx r_1 + r_2 \times \hat{f}(M)$ with regression error $\epsilon'(N)$. If the combined error of Fourier compression and linear regression $(\epsilon(N) + \epsilon'(N))$ does not exceed the user-defined threshold ϵ, then node I compresses $\hat{f}(N)$ into tuple (N, M, r_1, r_2) before forwarding it to the gateway. When the gateway receives this tuple it can approximate the time series of N based on $\hat{f}(M)$ with sufficient accuracy. To ensure that $\hat{f}(M)$ arrives at the gateway node intact, we update its entry in the cache of node I with a read-only flag. When node I finishes processing incoming traffic and forwards cached entries to the gateway, the read-only flag of $\hat{f}(M)$ prevents it from being modified at intermediate nodes.

Step B: If $\hat{f}(N)$ cannot be approximated as a linear function of another node's time series, $(\hat{f}(N), \epsilon(N))$ is cached in local memory. Node I forwards cached tuples to its parent, once it has finished processing incoming traffic.

C. Experimental evaluation

We have evaluated the performance of FC, FC-temporal and FC-spatial using real traffic data generated during a period of 14 days. In our simulations, we placed sensor nodes as in the Cambridge deployment and added a number of relay nodes to bridge disconnected network components. We set the communication range to 250m, and carefully placed gateway nodes to minimize the sum of hop counts from sensor nodes to the closest gateway. We then measured the communication cost of the three algorithms as we vary i) the user-defined error threshold and ii) the number of gateway nodes.

Figure 7 shows the total communication cost of FC, FC-Temporal and FC-Spatial in the simple scenario where we have only one gateway in the middle of the network. If data was

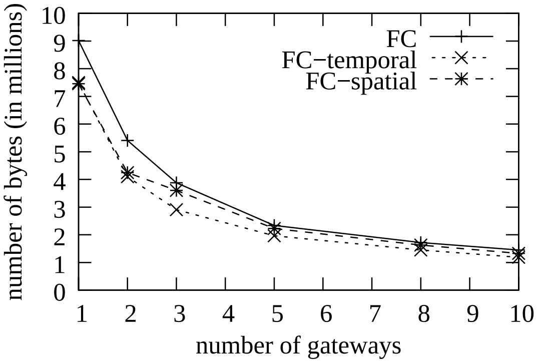

Fig. 8. Effect of varying the number of gateways on the communication savings of the proposed algorithms.

propagated in an uncompressed form, it would generate 32.3 MBytes of traffic. If the error tolerance is increased from 5 to 15 cars per 5 minutes, Fourier Compression (FC) yields communication savings that range from 44% to 90%. FC-temporal is 14%-30% more efficient than FC for tolerated errors of 5-15 cars per 5-min interval. Similarly, FC-spatial outperforms FC by 10%-35% for the same error range.

Figure 8 shows the effect of adding more gateway nodes on the total communication cost, assuming a fixed error threshold of 10 cars. The use of more gateway nodes results in better load balancing, and improves the performance of all three algorithms. This is anticipated since the more the gateway nodes, the shorter the paths that packets traverse to reach a gateway. FC-Temporal continues to yield significant communication savings compared to FC as we increase the number of gateways. However, the savings of FC-spatial diminish in networks with more than two gateways. As messages travel fewer hops to the gateway, they get fewer opportunities to be compressed along the way based on spatial correlations.

IV. Related Work

A lot of recent work has focused on exploiting relaxed precision requirements to compress data and reduce the communication load in the network. Lazaridis et al. [2] use a compression approach called piecewise constant approximation (PCA), a lossy compression scheme that represents a time series as a sequence of value-interval pairs (c_i, e_i), i.e. a constant value c_i during e_i epochs. Using PCA on the road traffic dataset would incur a high update frequency due to heavy fluctuations in the time-series (see Figure 5). Jain et al. [3] propose the use of Kalman filters, caching filter parameters that help predict the data instead of the static data itself. We plan to use this technique to propagate real-time traffic updates based on previous real-time and periodic updates. Deligiannakis et al. [4] construct a base signal from the node's time series characteristics and subsequently use it to approximate time series using regression. Their approach, which aims to exploit correlations between different time series on a single node, can be embedded in our framework as a replacement of Fourier and wavelet transforms.

Guestrin et al. [5] introduce the usage of kernel functions, which can efficiently compress data in regions of the network where significant spatial correlations are observed. In our traffic scenario, proximity of streams is not an indicator of high spatial correlation and therefore kernel functions cannot be fully exploited. Deshpande et al. [6] utilize Kalman filters to exploit temporal correlations, combined with probabilistic schemes to exploit spatial correlations between nodes. Their prototype, called BBQ, is a centralized pull-based system, which fails to detect outliers in the network. Ken [7] uses the same ideas as BBQ to compress data, but it is push-based, which means that source nodes proactively push data towards the gateway only when there is a need to update the gateway's model. Ken partitions the sources into clusters and examines spatial correlations within cluster boundaries. In contrast, our FC-Spatial algorithm does not only examine correlations between nearby nodes, as it seeks spatial correlations at each intermediate node of the communication tree.

Sadler et al. propose computationally-efficient compression algorithms for resource-constrained devices [8]. Unlike our work, they consider lossless compression and they do not exploit spatio-temporal correlations in data streams. Chou et al. exploit spatial correlations in [9], but, unlike our work, they use distributed compression techniques wherein each sensor compresses its data without knowing what the other sensors are measuring [10]. This technique requires the gateways to continuously predict an accurate structure of spatial correlations, which is not always possible in the case of traffic data.

V. Conclusions

We have presented an energy-efficient push-based approach to extracting traffic data periodically from a sensor network. Fourier-based lossy compression implemented locally at the sensor nodes was shown to offer 44-90% savings in communication costs for tolerated car-flow errors of 5-15 cars per 5-min interval. Furthermore, real traffic data obtained from traffic-monitoring sensors in the city of Cambridge were shown to exhibit significant temporal and spatial correlations. Our two novel algorithms, FC-temporal and FC-spatial, exploit such correlations to achieve significant benefits compared to Fourier compression alone. FC-temporal is 14%-30% more efficient than FC for tolerated errors of 5-15 cars per 5-min interval, whilst FC-spatial outperforms FC by 10%-35% for the same error range. Both algorithms are fully distributed and easy to implement in resource-constrained sensor and relay nodes. In the future, we plan to exploit consistent spatial correlations to identify redundant nodes, and optimize sensor placement. We will also investigate how to efficiently propagate real-time updates in order to provide users with timely information about unusual traffic events.

References

[1] http://www.moteiv.com.

[2] I. Lazaridis and S. Mehrotra, "Capturing sensor-generated time series with quality guarantees," in *ICDE*, 2003, pp. 429–442.

[3] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using Kalman filters," in *SIGMOD*, 2004, pp. 11–22.

[4] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Compressing historical information in sensor networks," in *SIGMOD*, 2004.

[5] C. Guestrin, P. Bodi, R. Thibau, M. Paski, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *IPSN*, 2004, pp. 1–10.

[6] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, “Model-driven data acquisition in sensor networks,” in *VLDB*, 2004.

[7] D. Chu, A. Deshpande, J. Hellerstein, and W. Hong, “Approximate data collection in sensor networks using probabilistic models,” in *ICDE*, 2006.

[8] C. Sadler and M. Martonosi, “Data compression algorithms for energy-constrained devices in delay tolerant networks,” in *SenSys*, 2006.

[9] J. Chou, D. Petrovic, and K. Ramchandran, “A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks,” in *Infocom*, 2003.

[10] D. Slepian and J. Wolf, “Noiseless coding of correlated information sources,” *IEEE Transactions on Information Theory*, vol. IT-19, no. 4, pp. 471–480, July 1973.