Dynamic clustering of multi-modal sensor networks in urban scenarios

Yicheng Wen a, Doina Bein b, Shashi Phoha b,∗

a Department of Mechanical Engineering, The Pennsylvania State University, USA
b Applied Research Laboratory, The Pennsylvania State University, USA

Abstract

The paper addresses the issue of self-adaptation of a multi-modal sensor network with mobile sensors to better observe and track events of interest in a changing urban scenario by presenting a software module (middleware) called Event-driven Network Controller (ENC) that resides at every sensor node in the network and is independent of the sensor type. ENC translates the requirements of the application layer into messages that are diffused locally with the purpose of clustering multi-modal sensor nodes in the vicinity of an event and dynamically changing the local network topology, all to enhance the quality of the multi-modal data fusion. ENC is implemented in NS-2 to show its applicability for tracking a mobile target in an urban scenario using a network of pressure, video, and magnetic sensors.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

A distributed sensor network consists of many low-cost sensor nodes that are networked to observe events that occur in a spatially and temporally distributed space. Observations, decisions and/or estimates made by individual sensors are fused together to create a composite view of the situation. Typically, the sensor nodes and the associated sensor network are resource constrained in terms of network bandwidth, node battery life, sensing range and communication radius. Consequently, to deliver robust sensor fusion performance, it is important to dynamically control and optimize the available resources.

Early protocols for tracking mobile targets by clustering nodes in sensor networks are proposed in [1–3]. The Dynamic-Space Time Clustering (DSTC) [1] algorithm groups homogeneous sensors in the space–time vicinity of an event to improve their observation and tracking. The authors of [4] present some design principles for middleware in wireless sensor networks, in which sensor nodes are clustered together to form a virtual machine that resides at the application layer; the middleware consists of a cluster control layer and a resource management layer. Our approach is different: the application layer resides at every node and it is the responsibility of the proposed middleware Event-driven Network Controller (ENC) to translate its requirements into messages to be broadcast in the network. A dynamic clustering algorithm based on Voronoi diagrams is presented in [5] where targets are tracked using an acoustic wireless sensor network. The Cougar project [6] is one of the earliest projects dedicated to storing the results of sensing in the sensor network itself. The sensors, organized as a static, wireless, grid-like network, are programmed through queries written in a high-level programming language (e.g. SQL) to collect data sets from themselves and adjacent sensors, select the relevant one, and decide where to store it in the network. Several query-based middleware projects for sensor networks were discussed in [7]. However, none of them addressed the issues of heterogeneity and scalability. The authors of [8] proposed a mobile agent-based middleware that uses the publish-subscribe mechanism to build the routes of interest. We consider instead mobile sensor nodes that are able to move to specified regions on demand, when requested. A simpler version of the proposed ENC middleware has been proposed in [9]; we present here a complete version with more experimental results.

An Event-driven Network Controller (ENC) is proposed that is a distributed, scalable, message-based middleware, residing at every sensor node in the network and responsible for self-organizing the network to the requirements of the targeted application. The ENC is validated by incorporating it into an NS-2 based software, that involves pressure, video, and magnetic sensors deployed in a simulated urban environment. Such models for self-organizing and self-adapting network topology are necessary for simulation-based design of urban sensor networks.

The paper is organized as follows. Preliminary notions for multi-modal data fusion are presented in Section 2. The major tasks of the ENC middleware are presented in Section 3. In Section 4 we present two weighted estimation methods for target location and...
we show how the ENC handles event tracking. In Section 5 we compare the ENC with the previously proposed DSTC [1] and we show that the ENC outperforms DSTC when both are employed to group sensors of the same modality for the same urban scenario. Section 6 summarizes these conclusions and gives directions for future work.

2. Multi-modal data fusion

When a mobile target approaches a sensor node and the perceived signal at the node is above a given threshold, an event is said to have been detected. The time-series data observed by a sensor of a certain modality can be compressed to a probabilistic finite state automaton (PFSA) over an a priori fixed alphabet [10]. The PFSA is designed to extract the maximum semantic information from the raw sensor data, while at the same time compressing the sensory data to minimize the network bandwidth requirements. At each node the data is handled in the Information Space Module (ISM).

During the training phase a number of PFSA of interest, called subpatterns, are identified and stored in the ISM of each sensor node. In the operational phase, when an event is observed by the sensor, the sensor’s ISM constructs a PFSA from the time-series data collected by the sensing unit and see whether it matches against the stored subpatterns. If successful, the ISM sends to the ENC the message subpatternObserved with the subpattern ID and the semantic distance [11] between the observed subpattern and the subpattern stored. The semantic distance [11] measures the signal deterioration from its origination to the sensor location; it was shown to increase as the target moves away from the sensor [11].

After receiving the message subpatternObserved from the ISM, the ENC broadcasts it as the message Subpattern after appending some control information (e.g., the node position, the node ID, and the time). At each node, the data received from sensors of different modalities is fused locally in the ISM in order to generate patterns of interest, called composite patterns.

A composite pattern is the Cartesian product of subpatterns in each modality. For $M > 1$ modalities, a composite pattern $P = \{P^1, P^2, \ldots, P^M\}$ is a $M$-tuple of subpatterns, one subpattern for each modality. The composite metric [11] is the semantic composite distance between a Cartesian product of the observed subpatterns and a composite pattern.

The flowchart in Fig. 1 presents how the clustering algorithm is executed and how data fusion occurs in a node that senses an event of interest.

We present next the major tasks of the ENC middleware, which extends the protocol DSTC [1] to include multi-modal and mobile sensors.

3. Dynamic clustering of multi-modal sensor nodes

The protocol DSTC [1] has five basic operations on a cluster: cluster formation, cluster disbanded, managing the cluster size by adding or removing nodes, and selecting a different cluster head within the same cluster. Our proposed ENC middleware extends DSTC to include the heterogeneity of the sensors and the self-adaptation of the network topology, due to the changing urban environment and the nodes’ mobility. It has two new operations: precluster formation and node re-positioning, and some operations of DSTC are modified to include the heterogeneity of the sensors. The full description of the messages exchanged between the ISM and ENC, or the ENC of two different nodes are presented in the appendix, Table A.1.

We say that a node is free when it is not part of a precluster or a cluster.

3.1. Precluster formation

When a sensor node detects a subpattern, the following steps are executed by the ENC in order to pursue a precluster formation.

1. After the ISM matches the constructed PFSA to a subpattern, it sends to the ENC the message subpatternObserved with the subpattern ID (SubPatternID) and the semantic distance (Metric).
2. On receiving the message subpatternObserved from the ISM, the ENC broadcasts it as the message Subpattern after it appends to it some control information such as the node position (NodePosition), the node ID (NodeAddress), and the current time (Time) (see Fig. 2a).
3. The ENC waits for messages Subpattern from other nodes.
4. When the ENC receives the message Subpattern from another node and has previously received a message subpatternObserved from the ISM, the ENC forwards the received message.
5. From its own and the received subpatterns, the ISM tries to decide on a composite pattern. If successful, the ISM sends to the ENC the composite pattern by the message patternObserved with the composite pattern ID (ComPatternID) and the composite metric (ComMetric) (see Fig. 2b).
6. If the ENC has received the message patternObserved from the ISM then it broadcasts the message createPrecluster with the information from that message and additional control information such as the node ID (NodeAddress) and the current time (Time). A timer of 10 s starts then at the ENC (see Fig. 2c).
7. After timeout, if the ENC has received the message patternObserved from the ISM and has received at least two createPrecluster messages from neighboring nodes, the ENC selects as the cluster head the node with the best composite metric among its own and the received composite metrics, and proceeds to the cluster formation.

3.2. Cluster formation

A cluster head is a node that holds the local maximum for the accuracy value of the sensed composite pattern of interest; in case of a tie, the node with the highest ID is selected.

Periodically, the ISM of any cluster member notifies the ENC about the observed pattern by sending the message patternObserved. For the rest of the paper, the terms PatternID, Metric, and Time refer to the fields of the last message patternObserved sent by the ISM to the ENC.

During a timeout of 10 s, the ENC waits for messages createPrecluster from neighboring nodes. After timeout:

1. If its own Metric < all received Metric values then the node proposes itself as a cluster head allowing the ENC to broadcast the message joinCluster (see Fig. 2d) and attempts to form a cluster.
2. If its own Metric > some received Metric value or the node did not decide on a composite pattern, then the node expects the message joinCluster from another node.

During the cluster formation, the selected cluster head attempts to form a cluster. The following steps are executed by the ENC of a node:

1. After broadcasting the message joinCluster, the ENC of the cluster head starts a timer of 8 s and waits for messages Hello from nodes that have decided to join its cluster.
2. If the node has chosen a cluster head, when the ENC receives the message joinCluster from that cluster head, the ENC replies by the message Hello (see Fig. 3a).
Fig. 1. Flowchart of the clustering algorithm and data fusion.

Table A.1
Messages exchanged between ISM and ENC, or ENC of different nodes.

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg_subpatternObserved(SubPatternID,Metric)</td>
<td>received from the ISM and the node is free</td>
</tr>
<tr>
<td>msg_subpatternRcvd(SubPatternID, Metric)</td>
<td>the ISM observes a subpattern, node is free and msgs Subpattern are received from other nodes</td>
</tr>
<tr>
<td>msg_patternObserved(ComPatternID,ComMetric)</td>
<td>is received from the ISM and the node is free</td>
</tr>
<tr>
<td>msg_joinCluster(ComPatternID,ComMetric, NodeAddress, Time)</td>
<td>received from the ISM and the node is free</td>
</tr>
<tr>
<td>msg_joinCluster retrieves from the cluster head OR the node is free, the ISM has received a msg moreNodeRequest from the ENC and the ISM sends moreNodeReply(ComPatternID,SubpatternID)</td>
<td></td>
</tr>
<tr>
<td>msg_moreNodeRequest(ComPatternID,Modality, IncreasedTR, NodeAddress, Time)</td>
<td>the ENC of the CH has not received enough msgs Hello OR the ENC has received moreData (ComPatternID,Modality) from the IS</td>
</tr>
<tr>
<td>msg_ObservedPattern (ComPatternID, ComMetric)</td>
<td>when the composite metric changes OR the ENC of a cluster member receives a msg observedPattern from the IS and a msg. Pattern from the cluster head, has a better metric and the Min_Lifetime of the cluster has expired</td>
</tr>
<tr>
<td>msg_leaveCluster(ComPatternID, Metric, NodeAddress, Time, NodeAddressDeleted)</td>
<td>the ISM of a cluster member periodically sends the observed composite pattern</td>
</tr>
<tr>
<td>msg_interference</td>
<td>the ENC receives a msg interference from the ISM</td>
</tr>
</tbody>
</table>
3. After the timeout, if the ENC of the cluster head has received enough messages Hello, then the cluster is formed. Otherwise the ENC broadcasts the message moreNode with the ID of the composite pattern (ComPatternID) and an increased transmission range (see Fig. 3b) and waits for more messages Hello.

4. A node that has sent its message Hello and has received the message moreNode, it ignores it.

5. A node that has not sent its message Hello and has received the message moreNode, its ENC sends the data fields of that message to the ISM by the message moreNodeRequest. The ISM will decide, based on that data, whether to join the cluster. In affirmative, the ISM indicates by a message moreNodeReply to the ENC to act on joining the cluster; ENC will send the message Hello to that cluster head (see Fig. 3c).

3.3. Cluster maintenance and data fusion

The cluster has a limited lifetime, called max_lifet ime, measuring the time elapsed after the cluster head has sent the first message joinCluster; during this time, the ISM of the cluster head does data fusion, periodically estimates the target location and velocity, and informs the ISM of the cluster members by broadcasting these...
estimates. As long as the ISM identifies a subpattern, it periodically sends to the ENC the message subpatternObserved, which is broadcast as the message Subpattern. In case the ISM ceases to send the message subpatternObserved, the ENC starts disbanding the cluster.

Whenever the ENC receives the message Subpattern from another node, it sends it to the ISM as the message subpatternRcvd. The ISM will fuse into a composite pattern the observed subpattern and the subpatterns received from other nodes, and it will send the composite pattern to the ENC as the message patternObserved. On receiving it, the ENC checks whether there is any change in the composite metric. In affirmative, the ENC broadcasts the message Pattern (see Fig. 4).

When the ENC of a cluster member receives the pattern from its cluster head, it will compare it against its computed pattern and may take actions (see Section 3.5).

3.4. Cluster size control

For the cluster head to be able to estimate the target location, a minimum number of cluster members are required. For example, the CPA-based algorithm (Section 4.1) requires at least three cluster members. Also, for a cluster to be able to detect a composite pattern, at least one sensor of each modality is needed to be member of the cluster. But if the cluster size is too large and the cluster members are too close to each other, then interference (e.g., low signal-to-noise ratio) can occur at the cluster head when it receives the messages Subpattern periodically from all its cluster members. To deal with enlarging or reducing the cluster size, the cluster head can send control messages.

If the ISM of the cluster head sends to the ENC:

- the message moreData asking for data of a certain modality, then the ENC broadcasts the message moreNode with an increased transmission range (IncreasedTR),
- the message interference indicating message interference, then the ENC sends the message leaveCluster to specific neighboring node(s), asking them to leave the cluster; the node(s) then leaves the cluster and becomes free (see Fig. 5a).

When the ENC of some non-cluster member receives:

- the message moreNode, then it sends to the ISM the message moreNodeRequest and expects the message moreNodeReply,
- the message moreNodeReply from the ISM, then the ENC sends the message Hello to the cluster head (see Fig. 5b).

3.5. Controlling the lifetime of a cluster

When the max_lifetime expires, the cluster head disbands the cluster by broadcasting the message leaveCluster; all cluster members, including the cluster head, become free. The cluster head can also disband the cluster when its battery power falls below a certain threshold. Also, when the ENC of the cluster head ceases to receive messages patternObserved from the ISM, then the ENC broadcasts a message leaveCluster and the node becomes free. Cluster members, on receiving a message leaveCluster from their cluster head, become free as well (see Fig. 6).

Since the target is mobile, it is possible that a node other than the cluster head may observe the pattern better, thus the current cluster may need to be disbanded and another cluster may need to be formed. Since there is latency in estimating the target location since the target has been detected, we propose the following lazy approach: the cluster will exist for at least some period of time called min_lifetime. During the min_lifetime, the cluster head does not change due to a better composite metric for the same composite pattern. Once the min_lifetime expires and before max_lifetime expires, it is possible for a cluster member or a non-cluster member to become a cluster head and force the current cluster head to disband the cluster.

When the ENC of the cluster head receives either the message createPrecluster, Pattern, or joinCluster, containing a better metric for the same pattern:

- If the min_lifetime has not expired, then the ENC ignores the message.
- If the min_lifetime has expired, then the ENC disbands the cluster by broadcasting a message leaveCluster. The cluster head and the cluster members become free and they can join the new cluster when the message joinCluster is received from that node.

If the ENC of some cluster member receives either the message createPrecluster, Pattern, or joinCluster, containing a better metric for the same pattern from a node other than the cluster head and:

- If the min_lifetime has not expired, then the ENC ignores the message.
- If the min_lifetime has expired, then the ENC informs the cluster head’s ENC that it will leave the cluster by sending the message goodbye. The cluster member becomes free and will join the new cluster if a message joinCluster is received from that node.

A node that continues to observe the pattern (i.e., the ENC receives the messages subpatternObserved from the ISM) and is part...
of a cluster, becomes free when it receives the message leaveCluster from its cluster head; subsequently it returns to the precluster formation step.

4. Target localization and estimation

The selected cluster head periodically collects measurements from its cluster members, fuses them and gives estimations of the target’s position and velocity. In this section, we present two target localization algorithms based on the semantic distance: the closest point of approach (CPA) tracking algorithm (in Section 4.1) and a probabilistic method (in Section 4.2). In Section 4.3, the target velocity is estimated by the cluster head based on the estimated location of the target at two consecutive points in time.

4.1. A CPA-based method to estimate the target location

The CPA tracking algorithm [3] was originally designed for an acoustic sensor field but can be extended to other types of sensors. An acoustic sensor can detect the loudness based on Doppler shift. When a target passes the closest point to the acoustic sensor, the sensory data signal reaches a peak and the event is signaled. The captured event is called a CPA event. The signal peak of a CPA event is directly related to the physical distance between the target and the sensor because the signal-to-noise ratio at the sensing unit decreases with the distance: the closer the target is to the sensor, the higher the signal peak of the CPA event is. To estimate the physical location of the target at a certain moment of time, the signal peak is used as a weight to triangulate the target position. For the above interpretation, the semantic distance is inversely proportional to the weight of a CPA event for the same type of sensors.

The position of the target is estimated by the cluster head based on the measurements from its cluster members to which weights based on the sensor modality are applied. We assume that each sensor is isotropic and has the ability to estimate the distance to the target. For example, pressure sensors give higher voltage readings when the target is right on top and lower readings when it is farther away. Such a voltage-distance characteristic curve can be used to estimate the distance once a voltage signal is sensed. Periodically, each sensor \( i \) that is part of a cluster generates a data point for estimating the target location; the data point is a tuple \( O_i = (t_i, r_i, w_i, x_i, y_i) \) where \( t_i \) is the timestamp of the measurement, \( r_i \) is the estimated distance to the target, \( w_i \) is the weight of the estimation, and \( (x_i, y_i) \) are the coordinates of the location of sensor \( i \).

Let \((x, y)\) be the estimated location of the target. We define the following cost function \( F \):

\[
F(x, y) = \sum_{i=1}^{n} w_i [(x - x_i)^2 + (y - y_i)^2 - r_i^2]^2.
\]  

(1)

Finding the optimal \((x, y)\) that minimizes \( F \) is a nonlinear least square problem. We note that at least three data points (i.e. \( n \geq 3 \)) are needed to eliminate the chance of having an infinite number
of optimal solutions. We use the GaussNewton method (Chapter 14 of [12]) to iteratively solve this problem. We select the location of the cluster head as the initial condition of the GaussNewton algorithm, since the cluster head is almost always the closest to the target (by the rule of clustering). In the rare case that multiple optimal points may exist, we would then choose the solution that is the closest to the cluster head. During the training phase, the weights $\mathbf{w}_i$’s can be selected based on the accuracy of the sensing. Sensors with more consistent readings (i.e., less variance) should have a higher weight than the ones with large variance. How to properly choose the weights is out of scope of this paper and is not addressed. In the simulations, all the weights are chosen to be 1.

4.2. A probabilistic method to estimate the target location

An alternative non-heuristic way to estimate the target location is to use a Bayesian approach. Let $C_i$ be a cluster of nodes that have decided on the same composite pattern $G$, represented by a set of atomic patterns $(G^m_{i,1})$ at the slow-time epoch $t \geq 0$; $G^m$ is the subpattern associated with $G$ in the modality $m$. When a multi-modal cluster has been formed, each sensor $i$ of some particular modality $m$ in the cluster $C_i$ has decided at time $t$ on a subpattern $G^m_i \in (G^m_{i,1})_1 \subseteq G$.

A data point generated by a sensor $i$ of modality $m$, located at the coordinates $(x_i, y_i)$ at time $t_i$ is a tuple $O_i = (t_i, x_i, y_i, e_i, m_i)$, where $e_i$ denotes the semantic distance computed by the sensor $i$. Let $d_i$ be the physical distance between the sensor $i$ and the target $T$; $d_i$ is upper bounded by the sensing range of the sensor $i$, denoted by $d^m$. The node $i$ is selected to be part of a precluster (and later on of a cluster) only if its $e_i$ is within a predefined threshold $e_i \in [0, e^m]$.

Let us partition the interval $[0, e^m]$ into $k$ subintervals, $A^m_i = [a^m_k, a^m_{k+1})$, where $k$ is the index of the partition and $a^m_0 < a^m_k$. We also partition the interval $[0, d^m]$ into $n$ grids $B^m_i = [b^m_n, b^m_{n+1})$, where $n$ is the index of the partition and $b^m_0 < b^m_n$. We can treat $e_i$ and $d_i$ as discrete random variables and we define

$$
Pr\{\{e_i \in A^m_i\}|\{d_i \in B^m_i\} = \text{the probability that the semantic distance } e_i \text{ of the sensor } i \text{ is in } A^m_i,
$$

given that the physical distance $d_i$ is in the interval $B^m_i$.

Let $S$ be the sensing region of the cluster $C_i$ beyond which a target cannot be detected by any of the cluster members. We define $S$ as the footprint of the cluster; $S$ is a relatively small region compared to the area covered by the entire sensor network. We partition $S$ into $K$ squares $\{l_j\}_{j=1}^K$, $K > 0$, such that the center of the square $j$ is $(l_b, l_b)$.

We make the following assumptions:

1. There is a single target within the footprint $S$ of the cluster.
2. The target triggers a dynamic cluster formation (i.e., there are enough nodes in the region to form a cluster) and the target does not move too fast (i.e., there is enough time for the cluster to form and the cluster head to get a chance of estimating the target location and velocity).
3. Each sensor is isotropic (i.e., the sensor measurement depends only on the distance between the target and the sensor itself and is not affected by the sensing angle).
4. The conditional probability $Pr\{\{e_i \in A^m_i\}|\{d_i \in B^m_i\}$ is the same for each atomic pattern $G^m_i$.
5. Given the real position of the target $T \notin I_b$, the events $\{e_i \in A^m_i\}$ and $\{e_i \in A^m_j\}$ are independent, for all $i \neq j$.

Given the measurements $\mathbf{O}_i$ from the cluster members, the distribution of the target location over the footprint $S$ is given by the following Bayes’ formula:

$$
\mathbf{w}_i \triangleq Pr\{T \in l_j|\\bigwedge_{i} \{e_i \in A^m_i\} = \frac{Pr\{\bigwedge_{i} \{e_i \in A^m_i\} | T \in l_j\} \cdot Pr\{T \in l_j\}}{\text{Normalization Factor}}
$$

$$
= \prod_{i} Pr\{e_i \in A^m_i\} | T \in l_j \cdot Pr\{T \in l_j\}
$$

$$
= \prod_{i} Pr\{e_i \in A^m_i\} | d_i \in B^m_i \cdot Pr\{T \in l_j\}
$$

Eq. (3) uses Assumption 5. Eq. (4) uses Assumption 3. The measurement of sensor $i$ depends only on $d_i$, which is the distance between the sensor $(x_i, y_i)$ and the location $(l_b, l_b)$.

The prior probability $Pr\{T \in l_j\}$ is assumed to have a uniform distribution over the footprint $S$ because, without the measurements, we know only that the target is within the footprint of the cluster. Thus $Pr\{T \in l_j\} = \frac{\text{Area}(l_j)}{\text{Area}(S)}$. The likelihood function $Pr\{\{e_i \in A^m_i\}|\{d_i \in B^m_i\}$ will be trained from the experiments conducted in the lab. Assumption 4 states that the conditional probability depends only on the subpatterns in the pattern library. Therefore the number of likelihood functions to be trained is equal to the number of subpatterns of our interest and is independent of the number of sensors in the sensor network. During an experiment, for each subpattern $G^m_i$, pairs of $(e, d)$ are generated and the relative frequency on $A^m_i \times B^m_i$ is used to define the corresponding probability function.

Definition 1. The estimated physical location $(x, y)$ of a composite pattern of interest $G$ sensed at time $t$ by the nodes in the cluster $C_i$ is defined as the conditional expectation over the squares $\{l_j\}_{j=1}^K$ of $S$:

$$
x = \sum_{l_b} l_b w_j \quad \text{and} \quad y = \sum_{l_b} l_b w_j
$$

where $w_j \triangleq Pr\{T \in l_j|\\bigwedge_{i} \{e_i \in A^m_i\} \}$.

4.3. Weighted method for velocity estimation

To estimate the target velocity, the timestamps of the messages Subpattern are used. During the timeout of $8$ s, the cluster head collects and sorts the received subpatterns in increasing order of their timestamps; then it divides them into two sets based on the median time – the median value among the collected timestamps. Two location-estimations and two time-estimations of the event are computed, based on the first half and the second half of the partition. These four estimations are finally used to compute the velocity of the event. The detailed steps are as follows:

1. The minimum timestamp begin_time and the maximum timestamp end_time are selected, the median timestamp median_time is computed; if the number of subpatterns is even, then the median timestamp is the average of the two middle values.
2. Based on median_time, the collected subpatterns are split in two sets: the set $S_1$ contains the subpatterns whose timestamp is $\leq$ median_time, and set $S_2$ contains the subpatterns whose timestamp is $\geq$ median_time.
3. Considering only the subpatterns in the set $S_1$, an estimated position $(\hat{x}_1, \hat{y}_1)$ is computed using the solutions to Eq. (1) as follows:

$$
\hat{x}_1 = \frac{\sum_{i \in S_1} x_i w_i}{\sum_{i \in S_1} w_i} \quad \text{and} \quad \hat{y}_1 = \frac{\sum_{i \in S_1} y_i w_i}{\sum_{i \in S_1} w_i}
$$

$$
\hat{x}_2 = \frac{\sum_{i \in S_2} x_i w_i}{\sum_{i \in S_2} w_i} \quad \text{and} \quad \hat{y}_2 = \frac{\sum_{i \in S_2} y_i w_i}{\sum_{i \in S_2} w_i}
$$

$$
\hat{x}_3 = \frac{\sum_{i \in T_1} x_i w_i}{\sum_{i \in T_1} w_i} \quad \text{and} \quad \hat{y}_3 = \frac{\sum_{i \in T_1} y_i w_i}{\sum_{i \in T_1} w_i}
$$

$$
\hat{x}_4 = \frac{\sum_{i \in T_2} x_i w_i}{\sum_{i \in T_2} w_i} \quad \text{and} \quad \hat{y}_4 = \frac{\sum_{i \in T_2} y_i w_i}{\sum_{i \in T_2} w_i}
$$
Similarly, considering only the subpatterns in the set $S_2$, an estimated position $(\tilde{X}_2, \tilde{Y}_2)$ is computed using the solutions to Eq. (1) as follows:

$$\tilde{X}_2 = \frac{\sum_{i \in S_2} x_i w_i}{\sum_{i \in S_2} w_i} \quad \text{and} \quad \tilde{Y}_2 = \frac{\sum_{i \in S_2} y_i w_i}{\sum_{i \in S_2} w_i}$$

4. Considering only the timestamps of the subpatterns in the set $S_1$, respectively in the set $S_2$, the weighted times $\tilde{T}_1$ and $\tilde{T}_2$ are computed:

$$\tilde{T}_1 = \frac{\sum_{i \in S_1} t_i w_i}{\sum_{i \in S_1} w_i} \quad \text{and} \quad \tilde{T}_2 = \frac{\sum_{i \in S_2} t_i w_i}{\sum_{i \in S_2} w_i}$$

5. The estimated velocity $\tilde{v} = (v_x, v_y)$ is computed as follows:

$$v_x = \frac{\tilde{X}_2 - \tilde{X}_1}{\tilde{T}_2 - \tilde{T}_1} \quad \text{and} \quad v_y = \frac{\tilde{Y}_2 - \tilde{Y}_1}{\tilde{T}_2 - \tilde{T}_1}$$

5. Experimental results

In this section we present an application of the proposed ENC middleware for tracking a mobile target in an urban multi-modal sensor network, simulated in NS-2, as shown in Fig. 7. Pressure sensors, video sensors (cameras), and magnetic sensors were deployed in a Manhattan-like grid; not all sensors are shown in Fig. 7. The large brown blocks represent the buildings in the city and a yellow dashed line is the road marking for a two-way street, 20 feet wide. In the ground of two-way streets we embedded 312 fixed pressure sensors. Each pressure sensor generates a digital voltage ranging from 0 to 1023, due to the pressure applied when it senses an object above. Video sensors, mounted on the buildings (four per building), have adjusted view angles so that the sensing coverage can be easily changed. A video sensor takes snapshots
of the scene within its view at some particular frequency. Video
sensors are used for classification, but not for localization. We posi-
tioned 72 magnetic sensors on both sides of the streets as well as in
the intersections on mobile platforms that can move freely along
the streets; they can be quickly relocated if necessary. Magnetic
sensors can detect the presence of large amounts of ferromagnetic
material. The purpose of the multi-modal sensor network is to
track any suspicious vehicle moving in traffic by clustering nodes
in its vicinity, shown in Fig. 7, on the left, center. A suspicious vehi-
cle triggers certain subpatterns on each modality (pressure, mag-
netic and video).

In the training phase, subpatterns were collected from sensors
detecting as a target a Segway RMP robot moving in the lab. The
subpattern $G_1$ models the voltage signature of the Segway RMP
passing by a pressure sensor. Subpattern $G_2$ corresponds to the
magnetic signature of the robot using a magnetic sensor. The sub-
pattern $G_3$ indicates the presence of the robot in the view of the vi-
deo sensor. The composite pattern $G$ is defined as a tuple of the
three subpatterns, $G = \{G_1, G_2, G_3\}$.

In the operational phase, we compared the performances of the
ENC and the DSTC for various trajectories of the Segway, at a con-
stant speed. The path parameter $p$ gives the ratio of the distance
between the target and the right side of the road and the width
of the street. For example, when $p = 0.5$, the target moves exactly
in the middle of the street. In our simulations, we range $p$ from 0.4
to 0.6 by increments of 0.02.

The ENC clusters the nodes observing the same composite pat-
tern $G$ and fuses the data from multi-modal sensors to localize
the Segway. The DSTC clusters the nodes observing the same
subpattern and collects the data from these single modal sensors
to localize the target; by running the DSTC, sensors of the same
modality track the target independently of the sensors of another
modality. (For example, the pressure sensors only use $G_1$ to observe
and track the Segway.)

Fig. 8a compares the number of multi-modal clusters formed
using the ENC and the number of homogeneous clusters using
the DSTC. We note that the number of single-modal clusters is
much larger than the number of multi-modal clusters and is
approximately equal to the number of multi-modal clusters times
the number of types of sensors. Fig. 8b shows the average size of
the clusters in both cases. The average size in case of the ENC is
3.89 while 1.59 for the DSTC. Recall that the CPA localization algo-

We analyze how the ENC performs under various choices of
$min\_lifetime$ and $max\_lifetime$: (1) $min\_lifetime = 30$, $max\_\ni-
time = 50$, (2) $min\_lifetime = 50$, $max\_lifetime = 80$, and (3) $min\_\ni-
time = 80$, $max\_lifetime = 120$ (all units are in seconds). The re-

![Fig. 9. Performance of the ENC when min\_lifetime and max\_lifetime vary.](image-url)
We also note that the average cluster size is roughly the same regardless of the values of the min\_lifetime and max\_lifetime (see Fig. 9b); this is due to the fact that the cluster size mainly depends on the topology of the network and the speed of the target.

Fig. 9c depicts the number of estimations. A longer lifetime keeps a cluster for a longer period of time so that it can perform more estimations of the target location.

The localization errors are shown in Fig. 9d. We note that, for a given speed of the target, there exists a best choice of the min\_lifetime so that the localization error is minimal. A shorter lifetime may disband the cluster too fast for the cluster head to collect enough data points for estimation. On the other hand, a longer lifetime allows a cluster to collect enough data points and make a better estimation but if the cluster stays for too long and the target moves out of its footprint, localization errors increase. We also recall that a longer min\_lifetime would prevent another cluster to be formed along the track of a target since nodes may not be free for forming another cluster.

We vary the target speed when min\_lifetime = 50 and max\_lifetime = 80. We ran the target at a constant speed of 0.04 m/s, 0.08 m/s, and 0.16 m/s, respectively, and a varying path parameter p. The results are presented in Fig. 10. As the target speed increases, the following observations are drawn:

- The number of cluster formed decreases due to the latency of cluster formation (Fig. 10a).
- The average cluster size increases since the target covers a wider region during the same period of time (Fig. 10b).
- The number of estimations decreases since the cluster does not have time to collect enough data points (Fig. 10c).
- The localization error increases (as expected) (Fig. 10d).

6. Conclusion and future work

We proposed a middleware called Event-driven Network Controller (ENC) that is responsible for clustering multi-modal sensor nodes only in the space–time vicinity of a moving object and improving the data fusion. We successfully apply the ENC to localize and track a Segway via the language-theoretic approach for multi-modal sensor data fusion. We showed that multi-modal clustering is more robust to the environmental changes than single-modal clustering. The ENC middleware is generic and can be used in other applications as long as the notions of subpattern and pattern are defined. The ENC is energy-aware and also is able to improve the quality of data fusion by clustering only the sensors in the immediate vicinity of an event and moving sensors to better observe events.

The ENC considers only one-hop clusters. In order to save energy, the sensor nodes may reduce their transmission power which may require them to form multi-hop clusters in order to group enough sensors of various modalities. We plan to extend the results of one-hop clusters to multi-hop clusters where the transmission
schedule of the nodes within the cluster plays a much more important role than it does in the one-hop cluster.

Additionally, analyzing the local traffic at a node can help the ENC to alter the transmission schedule based on patterns of communication detected in the past and stored locally at the node. We plan to design a traffic analyzer that observes the local traffic at a node in order to predict future events and act based on the most probable course of action.

Appendix A. Type of messages

see Table A.1.

References


