Data Fusion for Multimodal Traveler Information in a Wireless Environment

D. J. Dailey  
Assistant Professor  
(206) 543-2493

H. Xu  
Research Assistant  
(206) 543-5371

(206) 616-1787  
University of Washington  
Det. of Electrical Engineering, Box 35500  
Seattle, WA  98195-2195

M. P. Haselkorn  
Professor and Chairman  
(206) 543-2577  
fax (206) 543-8858  
University of Washington  
Dept. of Technical Communication, Box 352195  
Seattle, WA  98195-2195
ABSTRACT

This paper presents two aspects of data fusion for a multimodal traveler information application. The two aspects are: (1) system architecture and (2) analytical estimation of the state variable speed from the observed quantities Volume and Occupancy. The paper describes how the analytical estimation procedure is integrated into the overall architecture.

INTRODUCTION

In this paper, we describe a structured approach to developing an Intelligent Transportation System (ITS) data fusion application. We use this approach to develop a distributed computing application that is suitable for distributing dynamic data in real time to a large group of users. Our discussion focuses on two aspects of data fusion for traveler information. The first aspect is the communication architecture which provides support for multiple data sources and sinks. The second aspect is the specific data fusion algorithm, which estimates mean vehicle speeds using single inductance loops sensors. These two aspects of data fusion are combined and used to demonstrate how a specific algorithm is implemented in the context of the overall data fusion architecture of an ITS application.

The application under consideration was created and deployed in support of a wireless Multimodal traveler information system, the Seattle Wide-area Information for Travelers (SWIFT) project, a federal highway administration (FHWA) operational test (FOT). The paradigms used in developing the SWIFT data fusion application fit within the context of the FHWA ITS Architecture (1, 2).

COMMUNICATIONS ARCHITECTURE

Our development environment is composed of four software components that can be configured and linked to construct applications. The components constitute a toolkit that implicitly handles messages and communication, freeing developers to deal with the flow and interpretation of data streams. Each component is: (1) autonomous, i.e. components are distributed across computing platforms; (2) independent, i.e. components can operate in parallel; (3) a peer of any other component, i.e. components can be placed into an evolving hierarchical matrix with great flexibility; and (4) reusable, i.e. coding is efficient and much of the application design is reduced to configuring available components.

There are four types of components, each of which affects the flow of data. The component types are: (1) a Source Component, which makes a data stream available, (2) a Redistributor Component, which obtains a data stream from one component and redistributes it to one or more other components, (3) an Operator Component, which obtains data streams from one or more components and creates a new data stream for distribution to one or more components, and (4) a Sink Component, which obtains data streams from one or more components. Redistributors and Operators function both as client and server, while Sources are servers and Sinks are clients.

We use the existing network inter-process communication (IPC) facilities as a base for implementing the components that make up the building blocks of our application development environment. Unlike previous work that required extensive software for each participating operating environment, we require only a socket interface to an IPC library to implement applications on a wide variety of operating systems at low cost. Implicit in the use of the IPC services is the expectation that the appropriate guarantee of delivery and re-assembly is met. In the example presented here, the transmission control protocol/internet protocol (tcp/ip) is used to provide the transport layer services implicit in the use of IPC services. The use of a network-wide transport protocol implies that the application can be widely distributed geographically while guaranteeing connectivity. Problems associated with interoperability, compatibility, and addressing are invisible to the applications discussed here because the transport mechanisms supporting the IPC paradigm address these issues directly.

In the following sections, we describe each of our component types.
Source

A Source component exists for each sensor technology and makes the data stream from that technology available on the network. It does this by: (1) interfacing with the sensor technology, (2) using knowledge of the content of the data stream to build a self-describing data dictionary into that data stream, and (3) making the data stream available to other components on the network. Note that the Source component does not include the technology which generates the data stream but is a tool that allows developers to ignore the nature of that technology. Sensor technologies range from a single sensor with a single, simple data type to computer systems providing rich streams of complex data. Whatever the nature and complexity of the data-generating technology, the related Source component packages its stream of data to give it a peer status with other data streams on the network.

The Source component communicates directly with the sensor/system to obtain data and information about the format, contents, and meaning of the data stream. It builds data structures - to define format, contents, and meaning - that are provided to clients as an initial data dictionary before transmitting the sensor data. These constructs produce a self-defining data stream with an initial dictionary and following data. As the content of the information coming from the sensor changes in format, content, and structure, this component builds a new data dictionary and passes it on to “downstream” components.

Redistributor

A Redistributor takes data from one component and multiplexes it to other components. Each Redistributor is client to a particular data-providing component and allows the data stream from that component to “fan out” to each of its own client components. The Redistributor component has two basic functions: (1) scaling and (2) authorization.

In our structured development environment, scaling is achieved by linking Redistributors in a hierarchical fashion. Since each Redistributor services \( N \) clients, \( M \) layers of Redistributors service \( N \times M \) clients. Because components are autonomous, Redistributors can reside on any computer and operate in parallel. In this way, scaling can be economically achieved in fixed-size blocks through the addition of Redistributors operating on parallel computing platforms.

The Redistributor also performs authorization for data access. The Redistributor maintains a dynamic client list, and whenever new data is available, passes it on to clients on that list. When a request for data comes from a component not currently on the client list, a check is made against an authorization list. This authorization list is part of the initialization information for each Redistributor and may remain as simple as a static list or may include dynamic queries to a remote database. The dynamic or static authorization list and authorization mechanism are part of the configuration information. Similar authorization is performed by Operator and Source components as well.

Operator

The Operator component contains the data fusion activity that operates on the data content as it is passed from the input buffer to the output multiplexor. An Operator gets data streams from one or more components, performs functions on those data streams, and makes available a single, newly-created data stream to one or more components. Its general purpose is “data fusion,” converting data from the form provided by the serving components to a form desired by the eventual clients.

Each Operator component is associated with a specific data fusion activity. These activities can range from the combining of like types of data to the creation of new information from a variety of data sources. Like the Redistributor, the Operator also performs authorization for data access.

Sink

Our Sink component is the portion of our approach that is commonly viewed as an application by a user. It is the component that typically either has a user interface or performs an activity that affects the environment. It typically obtains data from one or more components (usually an Operator or a Redistributor which is downstream of an Operator).
In our approach, applications are constructed from components by connecting Sources, Redistributors, Operators, and Sinks in a hierarchical structure. We define an application to be composed of all of the structures necessary to take the original sources of data, perform data fusion, and distribute the resulting information to the data sinks. These structures include the individual component types, structural information about the path for data flow through the application, and information necessary to allow the Operators and the Sinks to use the data.

![Diagram](image)

Figure 1. SWIFT application in the context of TrafNet and BusView
The SWIFT Application

We combined instantiations of our four components to create the SWIFT data fusion application. Figure 1 shows the set of components used to build this data fusion application. In SWIFT, traffic congestion and probe vehicle information are used to estimate speed (and travel times) on freeway links, and that information is provided, in turn, to a commercial wireless carrier who broadcasts it to subscribers. These subscribers receive the information on three principle devices: (1) a Seiko Receptor™ watch, (2) a Delco car radio, and (3) an IBM portable computer. Each of the receiving platforms represents a potential value-added resale of the information. Figure 1 shows the collection of components that create the information source for the SWIFT data stream that ultimately feeds the three wireless delivery devices.

In Figure 1 there are two data sources: (1) inductance loop data from the Washington State Department of Transportation (WSDOT) in the top left and (2) automatic vehicle location (AVL) information for a fleet of transit coaches from the Metropolitan King County Department of Transportation in the top right. The data flows downward in the diagram, and each component modifies or adds information to the data stream. (For a more detailed description of the structures and applications in Figure 1, see “A Structured Approach to Developing Real-Time, Distributed Network Applications for ITS Deployment,” forthcoming in the ITS Journal.)

Within the context of the communication structure shown in Figure 1, the data fusion Operator (labeled “SWIFT Data Fusion Link Speed”) receives data from two components and creates a new data stream. The remainder of this paper presents a specific component of the data fusion performed by this Operator, namely the estimation of link speeds from volume and occupancy measurements.

DATA FUSION ALGORITHM

In this section we present an algorithm for estimating mean traffic speed using single inductance loop measurements of volume (counts of vehicle over a duration) and occupancy (the fraction of some total duration during which the inductance loop senses the presence of a vehicle). Mechanisms to estimate speed from single loops has been of interest to traffic engineers for some time as speed is not directly observable from single loop measurements (3, 4, 5, 6, 7, 8). Recent advanced traveler information system (ATIS) initiatives have created a requirement for a robust solution to this problem for a new class of applications, namely those which provide information to travelers. It is as a result of such an initiative (Seattle Wide-area Information for Travelers, SWIFT) that the present algorithm was formulated.

In this section we acknowledge the statistical nature of the measurements made using inductance loops and present an algorithm to estimate speed that accounts for the statistical nature of the estimate as well as providing a robustness test for our estimate. There are four measurements that are made by a traffic management system, Volume $N_i$, Occupancy $O_i$, speed $s_i$, and vehicle length $l_i$ (but only volume and occupancy are available from single loops). These measurements are, by their nature, realizations taken from the probability distributions of the underlying variables at the time the measurement are made. Observations of these variables are typically combined to create estimates of speed; for example, several authors have used a ratio of volume ($\hat{s}$) and occupancy ($\hat{g}$) to estimate speed $s = s/\hat{g}$ (3, 4, 9, 10). ATIS efforts typically require estimates of speed and travel times but rely almost completely on the measurements made by traffic management systems, and, as such, require the use of single inductance loop speed estimates.

Previous work has not explicitly included the statistics of the estimated quantities when making estimates of variables that are not observable. This work explicitly considers the statistics of estimates created using observations from traffic management systems. The typical measurements are volume ($N_i$) and occupancy ($O_i$), and the relationship between volume, occupancy, speed $s_{ij}$, and length of the $j$th vehicle is $l_{ij}$,

\[
0 = \frac{1}{T} \sum_{j=1}^{N_i} l_j / s_{ij}
\]

where $T$ is the duration of the measurement. The speed and vehicle length are random variables with mean values and statistical distributions. We can express this by writing the speed and length observations as the expected value (mean) and some deviation ($\Delta l_j, \Delta s_j$) that occurs for this observation,
\[ l_{ij} = \bar{l} + \Delta l_{ij} \]
\[ s_{ij} = \bar{s} + \Delta s_{ij}. \quad (2, 3) \]

Combining these terms in the form of the RHS of equation (1), we get
\[ \frac{l_{ij}}{s_{ij}} = \frac{\bar{l}}{\bar{s} + \Delta s_{ij}} + \frac{\Delta l_{ij}}{\bar{s} + \Delta s_{ij}}, \quad (4) \]

where the statistics of the deviation term are selected such that \( E\{\Delta l_{ij}\} = E\{\Delta s_{ij}\} = 0 \) and \( E^* \) is the expected value operator.

Each measurement produces a pair of volume \( (N_i) \) and occupancy values \( (O_i) \). To use the statistics of these measurements, letting \( E_i \) denote the conditional expectation over all realizations that have the volume \( (N_i) \), the conditional expected value of equation (1) is
\[ E_i\{O_i\} = \frac{N_i}{T} E_i\left\{ \frac{l_{ij}}{s_{ij}} \right\}. \quad (5) \]

Inserting equation (4) in (5), we get
\[ E_i\left\{ \frac{l_{ij}}{s_{ij}} \right\} = E_i\left\{ \frac{\bar{l}}{\bar{s} + \Delta s_{ij}} + \frac{\Delta l_{ij}}{\bar{s} + \Delta s_{ij}} \right\}. \quad (6) \]

Rearranging the RHS, and recognizing that \( E\{\Delta l_{ij}\} = 0 \), we get
\[ E_i\left\{ \frac{l_{ij}}{s_{ij}} \right\} = E_i\left\{ \frac{\bar{l}}{\bar{s} + \Delta s_{ij}} \right\} = \frac{\bar{l}}{\bar{s}} E_i\left\{ 1 + \frac{\Delta s_{ij}}{\bar{s}} \right\}. \quad (7) \]

Expanding the RHS in a power series, we get
\[ E_i\left\{ \frac{l_{ij}}{s_{ij}} \right\} = \frac{\bar{l}}{\bar{s}} E_i\left\{ 1 - \frac{\Delta s_{ij}}{\bar{s}} + \frac{\Delta^2 s_{ij}}{\bar{s}^2} - \frac{\Delta^3 s_{ij}}{\bar{s}^3} + \ldots \right\}. \quad (8) \]

Noting that \( E\{\Delta s_{ij}\} = 0 \), approximating the series with three terms, and inserting the result in equation (5), we get
\[ E_i\{O_i\} = \frac{N_i}{T} \left[ \frac{\bar{l} \left\{ \frac{\Delta s_{ij}^2}{\bar{s}^2} \right\}}{1 + \frac{E_i\{\Delta s_{ij}^2\}}{\bar{s}^2}} \right]. \quad (9) \]

The variance of the speed estimate can be written \( \sigma_s^2 = E \{ \Delta s_{ij} \} \), and by substituting and rearranging, we get,
\[ N_i = \frac{sT}{\bar{l}} E_i\{O_i\} \left[ \frac{s^2}{\sigma_s^2 + s^2} \right]. \quad (10) \]
The measurement of the occupancy is also a random variable with some mean and some deviation from that mean for the $i$th measurement. We can express this as

$$O_i = \bar{O} - \Delta O_i \quad \bar{O} = E\{O_i\}.$$ (11)

Substituting (1) into (10), we get

$$\frac{N_i}{O_i} = \frac{\bar{O}}{\sigma_i^2 + \bar{O}^2} + \frac{\Delta O_i}{\sigma_i^2 + \bar{O}^2}.$$ (12)

This form has a deterministic component containing only moments of the speed distribution and a stochastic component containing $\Delta \sigma_i$. We next consider the solution of the deterministic component.

Measurements from a traffic management system are realizations from statistical distributions. To address the variability of the observations, we present a filtering approach. The general form for the dynamics and observer equations for a Kalman filter are (11)

$$X_{k+1} = G_k (X_k) + w_k$$

$$Z_k = h_k (X_k) + v_k.$$ (13, 14)

For the $k$th time step we select our state variables to be the estimate of speed for the last two time steps. This autoregressive-like approach explicitly identifies a temporal correlation between speed estimates and recognizes that $\bar{O}$ has some inherent variation in addition to the noise component. For our observables, we use the ratio of the measurements for the two previous time steps. We also note that in equation (12) there are deterministic and stochastic components, and we use the deterministic portion to construct the measurement function

$$X = \begin{bmatrix} \bar{O}_k \\ \frac{O_k}{N_k} \\ \frac{O_{k-1}}{N_{k-1}} \end{bmatrix}, \quad Z = \begin{bmatrix} \frac{O_k}{N_k} \\ \frac{O_{k-1}}{N_{k-1}} \end{bmatrix}, \quad H_k (X_k) = \begin{bmatrix} \frac{\sigma_k^2 + \bar{O}_k^2}{\bar{O}_k^2} \\ \frac{\sigma_{k-1}^2 + \bar{O}_{k-1}^2}{\bar{O}_{k-1}^2} \end{bmatrix}.$$ (15)

where the measurement equation for $h_k (X_k)$ is nonlinear in the state variables. The linear Kalman filter equations are written (11)

$$X_k = G_k X_{k-1} + w_{k-1}$$

$$Z_k = H_k X_k + v_k.$$ (16, 17)

where the measurement equation is a linear function of the state variables. To use the linear filtering result, we adopt the extended Kalman filter approach, which linearizes the measurement equation from (14) about a point $X_k^p$ (for implementation we select this point to be the last $X_k$)

$$h_k (X_k) = h_k (X_k^p) + dh(X_k^p) (X_k - X_k^p)$$ (18)

and create a new measurement equation,
\[ \hat{Z}_k = \hat{H}_k X_k + v_k , \]  

(19)

where

\[ \hat{Z}_k = \left\{ Z_k - h_k \left( X_k^p \right) + dh \left( X_k^p \right) X_k^p \right\} \quad \hat{H}_k = dh \left( X_k^p \right) \]  

(20)

and

\[
dh_k (X_k) = \begin{bmatrix}
- \frac{3\bar{I}}{T} & \frac{\bar{s}_k^2 + \sigma_s^2}{\bar{s}_k^4} \\
0 & 0 \\
- \frac{3\bar{I}}{T} & - \frac{\bar{s}_k^2 + \sigma_s^2}{\bar{s}_k^4} 
\end{bmatrix}.
\]  

(21)

Our state-transition matrix \( G \) provides weights for the contribution of \( \bar{s} \) from the previous two time steps,

\[ G_k = \begin{bmatrix} a \\ b \\ 1 \\ 0 \end{bmatrix}, \]  

(22)

where \( a \) and \( b \) are based on the temporal correlation of \( \bar{s} \). The noise contributions are

\[ Q_k = E \left\{ w_k w_k' \right\} \quad R_k = E \left\{ v_k v_k' \right\} \]  

(23)

where

\[ Q = \begin{bmatrix} \sigma_s^2 & 0 \\ 0 & \sigma_s^2 \end{bmatrix} \quad R = \begin{bmatrix} \frac{\sigma_v^2}{\bar{v}} & 0 \\ 0 & \sigma_v^2 \end{bmatrix}. \]  

(24)

With these definitions, we can use the linear filter solution,

\[
P_k^1 = G_k P_{k-1} G_k^T + Q_{k-1} \]

\[ K_k = P_k^1 \hat{H}_k^T \left[ \hat{H}_k P_k^1 \hat{H}_k^T + R_k \right]^{-1}, \]  

(25, 26, 27, 28)

\[ P_k = P_k^1 - K_k \hat{H}_k P_k^1 \]

\[ X_k = G_k X_{k-1} + K_k \left[ \hat{Z}_k - \hat{H}_k G_k X_{k-1} \right] \]

from reference (11) to update the state variables at each time step. This provides an algorithm to create a maximum-likelihood estimate of the speed using the observed volumes and occupancies. The robustness of this estimate can be tested by calculating the mean car length for each estimate using

\[ \bar{l}_i = N_i \frac{Q_{i,T}}{N_i} \left[ \frac{\bar{s}_k^2}{\bar{s}_k^2 + \bar{s}_k^2} \right]. \]  

(29)

and comparing this estimate with long time estimates of the mean \( \bar{l} \) and standard deviation \( \sigma_l \) of the length distribution. If \( c < \bar{l}_k < d \) (where \( c \) and \( d \) are selected based on the statistics of \( \bar{l} \)), the speed estimate is deemed to be robust.

CONCLUSION
This paper presents an algorithm to estimate speed from single inductance loops in support of data fusion. This algorithm is placed within the context of our communications architecture used to create ITS applications. This communication architecture provides a scalable, reusable methodology to create and distribute traveler information. The analytical fusion algorithm specifically acknowledges the statistics of the problem and uses the statistics of one of the observables to set criteria for evaluating the reliability of the estimate. The analytical fusion algorithm is implemented within an Operator component within the overall communication architecture. This combination of communication architecture and data fusion provides a reliable data stream to the wireless multimodal traveler information systems created by the SWIFT project.

ENDNOTES