Runtime Simulation for Post-Disaster Data Fusion Visualization

Thenkurussi Kesavadas*
Virtual Reality Laboratory and Center for Multisource Information Fusion (CMIF)
The State University of New York at Buffalo
Buffalo, NY 14260
USA

kesh@eng.buffalo.edu

ABSTRACT

With regard to the threats of recent natural and man-made disasters, it is important for a control center to be aware of the situation and be able to assess the threat. However, simulating a large amount of post-disaster fused data is a complicated task, and its visualization is even more difficult to achieve with the paradigm of common geo-referencing systems. We have developed a post-disaster monitoring interface that runs in a fusion-based simulation with High Level Architecture/Run Time Infrastructure (HLA/RTI). In our visualization system, damage and recovering activities are presented in a fast GIS vector map with convenient data and display manipulation. All data that comes from the data fusion federates is displayed at run-time and stored for further analysis. In addition, the pattern of time-aggregated data has enabled dynamic visualization, which includes the morphing of the casualty clusters. This feature provides an effective way to keep track of a region so that a user can easily be aware of the emerging trends. A unique approach to multiple views by the integration of 2D and 3D displays of the fused data is also described.

KEYWORDS

Disaster, emergency response, situation awareness, fusion, integrated architecture, network visualization.

1. INTRODUCTION

The time immediately following a natural or man-made disaster can be a chaotic experience to any individual or community. This is evident with regards to the natural and man-made disasters, which have occurred, in recent years. A prior study has shown that in an earthquake situation, the information collected and dispersed in the first 72 hours is the most crucial, since most people still severely injured after this time are not likely to survive [1]. When a disaster spreads over an area, and causes thousands of casualties in a short time, it is nearly impossible to manage the disaster by human observation alone because of the massive amount of incoming information. In fact, for large-scale disaster management, the first and most imperative step is the awareness of the situation in order to optimize the allocation of available resources. Therefore, the situation awareness is an essential role of a disaster monitoring or visualization system. With the paradigm of conventional geo-referenced display, this is difficult to achieve because its implementation is limited to the positioning of the corresponding graphic images. This usually results in thousands of scattered and cluttered icons on the display. The present research provides a monitoring environment through efficient data management and a user-friendly graphics interface that deals with the massive influx of data from the data fusion process. Furthermore, our technology adds time-aggregated data management that contributes to the visualization of more abstract and comprehensible graphics. This approach encourages tactical thinking and strategic control for a severely attacked area [2].

* Author for communication

Kesavadas, T. (2006) Runtime Simulation for Post-Disaster Data Fusion Visualization. In *Visualising Network Information* (pp. 16-1 – 16-14). Meeting Proceedings RTO-MP-IST-063, Paper 16. Neuilly-sur-Seine, France: RTO. Available from: http://www.rto.nato.int/abstracts.asp.

1.1 Previous Works

Recently, researchers have been involved in data fusion for dealing with complex natural phenomena or algorithmic problems. A few visualization projects exist involving information fusion, such as a weather visualization application for emergency planning [3], NASA's wind tunnel simulation [4], and seismic activity visualization by the University of California at Irvine [5]. A research team from the University at Buffalo has also been working to create a battlefield visualization scenario using fused data [6].

Without fused data, there have been studies on emergency response, such as decision making aids by interaction through voice/gesture recognition completed by Pennsylvania State University [7], a framework for incorporating the many emergency response models for a simulation by Jain and McLean [8], and satellite/airborne image or video processing for the Kocaeli earthquake in Turkey by Ozisik and Kerle [1]. The works mentioned above implemented data processing and mapping in a two or three dimensional space, but do not provide abstract information from which a user could comprehend a situation. The work of Kim and Kesavadas considered effective icon/symbol generation regarding the viewer's visual recognition, which has been a cognitive issue in the military community since the advent of the digital display in the 1960's [9]. They have suggested *Automated Dynamic Symbology* by parameterization of graphic components connected to fused data. Their methodology was considered as an appropriate feature for visualization of strategic information and remote-networked implementation, and served as a basis in the current research.

1.2 Issues and Our Approach

The present work is an achievement of a large scale fusion-based post-disaster simulation project. For simulation test, it specifically takes the post-earthquake situation data which occurred in Northridge, California on January 19, 1994. This was considered to be one of the worst earthquakes in the Los Angeles area in recent memory [10]. Our simulation model relies on the output from HAZUS [11] developed by the Federal Emergency Management Agency (FEMA). The fused data is currently being produced by a multidisciplinary group at our institution [12, 13]. The fusion output includes data of low-level fusion (identification) which covers roadway damage, casualties, hospitals, and ambulance routing and police information. As mentioned earlier, this approach usually causes information overflow to a viewer, such as different types of icons cluttering and overlapping on top of each other. Therefore, our system includes high-level fusion data (situation awareness and threat assessment), such as casualty clusters and its trends and prediction.

A challenge for the visualization of a post-disaster simulation is to deal with the substantial and complex data interface so a user can manipulate and retrieve desired information. Unlike the previous works, our application provides a visual display of emergency response data at run-time and through a networked simulation environment. The advantage is that it displays information as it is received and without delay. Commercially available Geographic Information System (GIS) software has been used for the data construction of urban terrain and traffic network. However, they are only capable of low-level fusion output, which is the functionality of location and identification. They do not have the functionalities for high-level fusion that demands many complicated tasks, such as large data-set manipulation, dealing with time aggregated data, and the capability of putting depth and height cues to the display.

To deal with such enormous and complicated data, we have developed the common class interface for sharing and synchronization of 2D and 3D graphics. It allows a user to see what resources are available, and where casualties are located in 2D, as well as 3D, which give a better understanding of the spatial relation amongst the resource objects. Each visual mode adopts different application programming interfaces (API) and rendering environment, Windows MFC and OpenGL for example. Our work achieved implementation of both in one application combined in sync, which is described in the following sections.

2. RUN-TIME FEDERATION INTERFACE

In this project, information communication is implemented with the HLA/RTI composed of several federates. A federate simply being one piece of the RTI which carries out a specific task, such as information regarding walk-in casualty or medical facility. The RTI allows for common variables to be changed by one federate and then updated in another based on the concept of "publishing" and "subscribing" to variables [14]. The following section explains the interface of the post-disaster simulation and describes how the data is passed to the visualization system.

2.1 The Federation Interface of HLA/RTI

The importance of HLA/RTI in the current simulation is that all simulation data is being generated at run-time. It should also be noted that for each federate it is crucial to know the current state of the situation at all the times. This is achieved through the report flow, or exchange interface, between federates (Fig. 1).

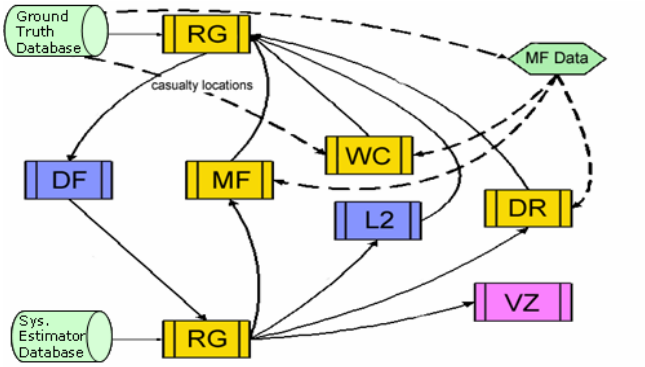

Figure 1. Post-earthquake simulation data flow in the HLA/RTI network.

The Report Generator federate (RG) generates all reports for the simulation based on the output from the HAZUS earthquake model [10]. Data Fusion (DF, also called Level-1 fusion), then decides which reports are not repeated and fuses them into one report. This information provides the core to the simulation which the rest of federates use to carry out their own tasks. Level-2 fusion (L2) determines the time-stamped formation of casualty clusters from the casualties reported by RG. The figure shows the interaction between other federates, such as Walk-in Casualty (WC), Medical Facility (MF), Dispatcher and Router (DR), and Visualization (VZ).

2.2 Data Interface of the Visualization Federate

Unlike the data relay in other federates, the visualization federate currently only takes information from the rest of the federation via Report Generator/Estimate Director. To create a highly abstract and robust runtime performance, we adopted the C# (C sharp) programming language on the Microsoft .NET platform [15]. Because the HLA/RTI is designed to support only C++ objects, it was critical to come up with a technique in order to integrate the two different programming codes. A solution was found for bridging the two executions not on the programming level, but the OS level. A system was devised for a directory to be setup where the C++ process stores all the report messages in ASCII format which can then be read by the C# process. Running in a call back loop, the directory watcher notifies the parser immediately after it gets the reports from the estimate director federate. The stored ASCII text is then parsed and stored in a shared memory for synchronization of the 2D and 3D graphics (Fig. 2).

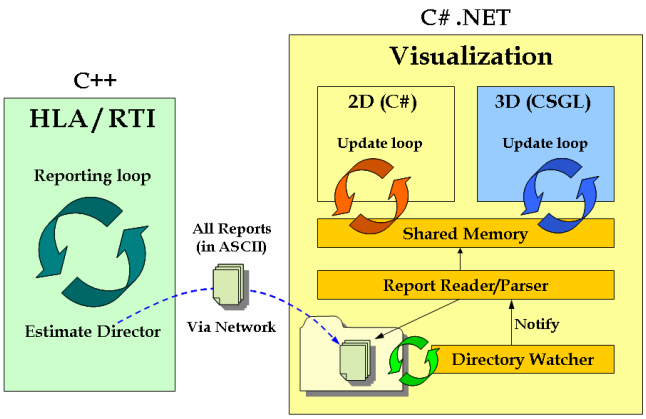

Figure 2. Data interface of the visualization federate between the HLA/RTI architecture.

3. MODELING OF VISUALIZATION FRAMEWORK

The following section addresses how we achieved efficient data manipulation of a large data set and integration of two different kinds of graphics (2D and 3D) to provide sufficient options to a viewer.

3.1 Data Layers: The Visualization Pipeline

Layering is a useful way to organize massive GIS data. The United States Geological Survey (USGS) [16] offers an accurate depiction of the Northridge area which was used for our vector map. In addition to map viewing capabilities, a monitoring capability has also been developed. All the GIS data is layered at the bottom of the pipeline and rendered first. Built on top of this information is the federate data from the HLA/RTI. Placed on top of the federate information is the graphical user interface (GUI) developed for the simulation (Fig. 3). Data layering and implementation not only provided an easy-to-debug environment to programmers, but also produced better rendering performance.

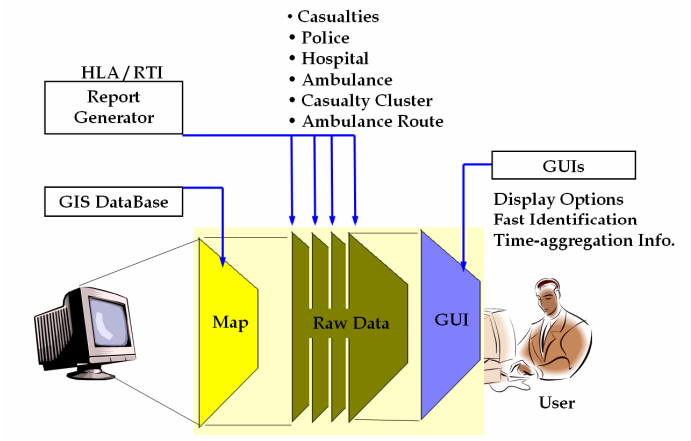

Figure 3. Data layers of the software architecture

3.1.1 Map Layer: Generation of Fast Vector Map

Two C# namespaces were created to handle the generation of the GIS map database for the display (Fig. 4). First, the *Geometry* namespace deals with the definition of all geometrical entities in a hierarchical structure. Another namespace, *GIS,* stores structured information of GIS objects for display. It uses a primitive class definition from the *Geometry* namespace with real GIS data. While all other GIS data was used from *.dxf files, a detailed road network is generated from a Tele-Atlas database file [17] which is helpful in determining ambulance routes and retrieving street information. Each of the links is represented by a 256 character line containing information like Link ID, Length, Street Name etc., which is further divided in a database of start and end points. Each end point is known as a node, which is stored with a unique ID and Universal Transverse Mercator (UTM) geographic coordinate in the database. A *road link* class then stores all the information in runtime memory. Finally this map is stored in the *Common Container* class structure for further process.

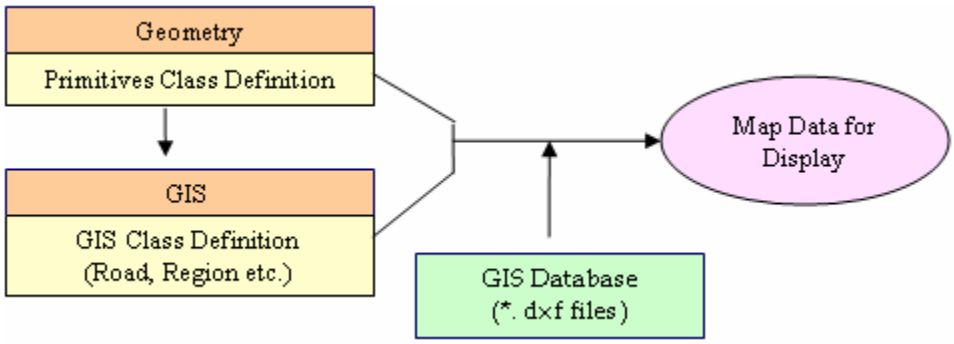

Figure 4. Pipelines to Map Data Generation.

3.1.2 Raw Data Layer: Data Extraction from Incoming Reports

The Disaster-RTI defines all types of federate class definitions including casualties, police, ambulance, medical facility, roadway damage, ambulance route etc., in terms of their properties and functions (Fig. 5). Dynamic arrays have been built on top of this information to store their objects and reports in a hierarchical order. A separate class called *Visualization* is then defined for automatic generation of different color data which is used for each zip-code region in the Northridge area. This class is also used for identifying proper symbols for each federates' database which is stored in a symbol database.

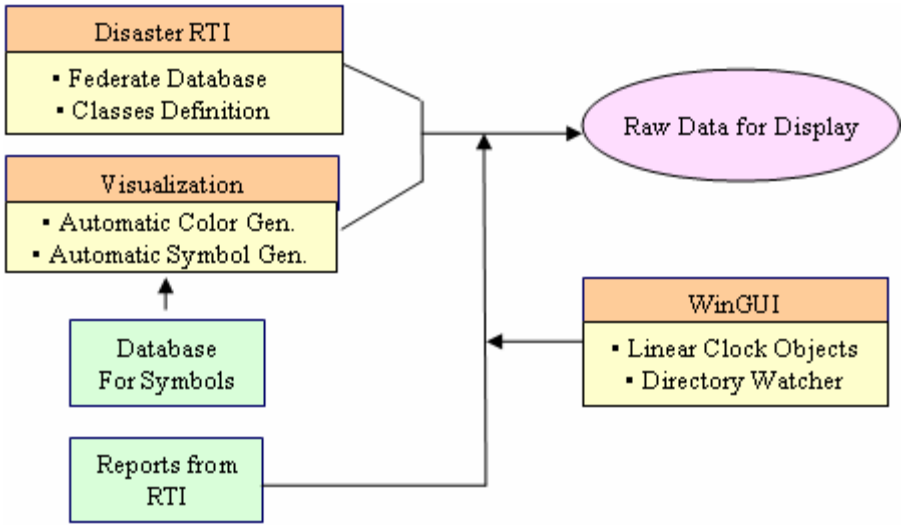

Figure 5. Pipelines to Report Data Generation

The directory watcher class and its functions are defined in the *WinGUI* namespace which keeps a track of new reports in the directory folder. For every report from the RTI, raw data is generated for display by combining these files and having the data stored in the *Common Container* class for further processing.

3.1.2 GUI Layer: User Interface for Data Manipulation

Figure 6. Linear clock and track bar for the manipulation of time-aggregated data.

What is controlled by the GUI is directly related to what is being captured in the federate layer underneath. The GUI includes a menu and tool bar control system with the capabilities of mouse and keyboard interactions. Since the simulation has been designed to run over a period of time, it is important to store accumulated federate data for further usage. The track bar shown in the GUI pane gives the user the ability to go back in time (Fig. 6). This way a person can see what progress is being made in specific areas over a period of time. It also has an option for time scope expansion, which allows for a longer duration of time to be rendered at once [18].

3.2 Integrated Simulation Architecture

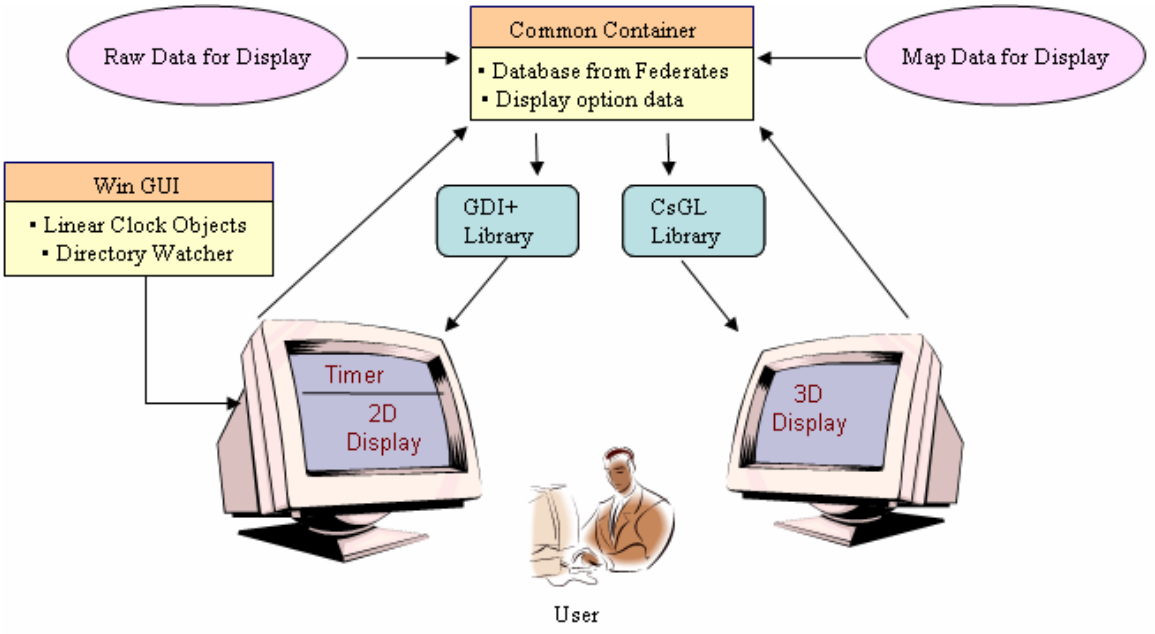

Figure 7. Integrated simulation architecture for multiple and different display of 2D and 3D.

We took advantage of multiple windows of two different graphics modes: two dimensional Windows MFC-based graphics, and three dimensional OpenGL-based graphics. For the synchronization in these multiple windows, display variables in either the 2D or 3D window need to be updated in the other window. Since it is impossible to swap references on two files due to restriction on circular dependency in C#, we developed a common class to access the shared memory and update the value (Fig. 7). All the corresponding objects for the federate report database (raw data) and map data are stored in the *Common*

Container which can be passed to the 2D and 3D windows for display as needed. Using the Graphical Device Interface Plus (GDI+) library for creating graphics in C#, all 2D graphics could be created in an effective manner [19]. The 3D display objects and navigation controls were then defined in the *3D display* class using the CsGL graphics library, which is simply an OpenGL wrapper for the C# programming language.

4. RUN-TIME FUSION DATA VISUALIZATION

The following sections report on the user interface and visual displays for low level fusion functionality (position and identification) and high level fusion functionality (situation awareness).

4.1 Multiple Display: 2D Fast Vector Map & 3D Visualization

The present research includes graphics controls and data manipulation found in common geo-referencing system. The zip code areas, road data and even the colors used, can be altered as needed by the user. This allows for the simulation to be viewed in a way that is least visually distracting so more attention can be focused on the resources (Fig. 8).

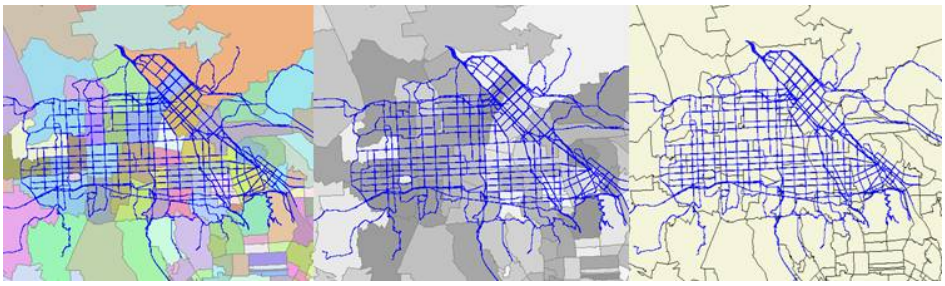

Figure 8. Manipulation of map environment: color (left), grayscale (middle), and single color.

The image of Fig. 9 (left) shows raw data from the report generator, such as casualties, police and ambulances. The image in the middle shows the corresponding view in 3D space. Even though 3D simulation has some disadvantages, such as unfamiliar interface and viewpoint control, the height and depth cues are an invaluable source of knowledge to a user [20]. This is especially true in a natural or man-made post-disaster simulation for an urban area. Height and depth cues are crucial for buildings or other volumetric spaces. As mentioned, this 3D view works in sync with the 2D graphics, so that viewers can easily switch their attention at their own preference. In the future the 3D view could be integrated with 3D building, structures and landmarks and to provide more comprehensive urban casualty visualization.

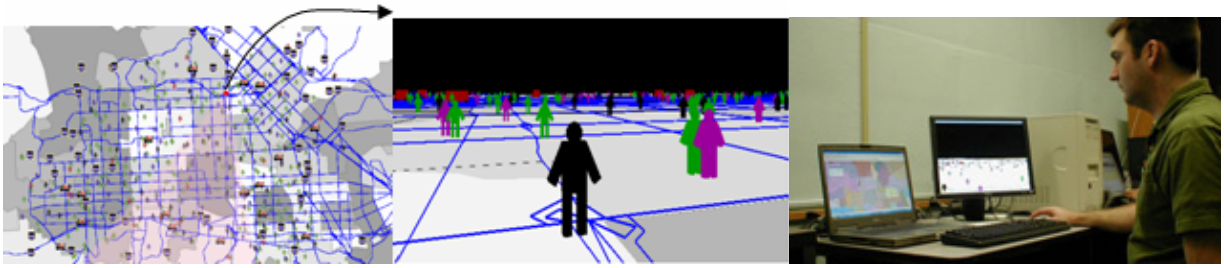

Figure 9. 2D view of Northridge Area (left), corresponding 3D view (middle), and actual implementation of multiple display visualization system.

4.2 Display Control

The requirement of fast identification is essential for visualization of low level fusion. In order to be able to view only federate data which is desirable, a menu bar system was developed to switch certain data layers 'on' and 'off'. As mentioned, this avoids the problem of cluttering that occurs if too many reports are coming into the visualization federate at one time. A context menu has been included which can be accessed by right clicking in the display window and performs the same function as the menu bar (Fig. 10, left). In the 3D view, an independent viewing position was created to allow for an outside viewer to navigate throughout the 3D environment to give a better sense of realism. We took advantage of the ability to change the scale and size of the icons in 3D, so based on the size of the data, a user can get a better understanding of the disaster scenario.

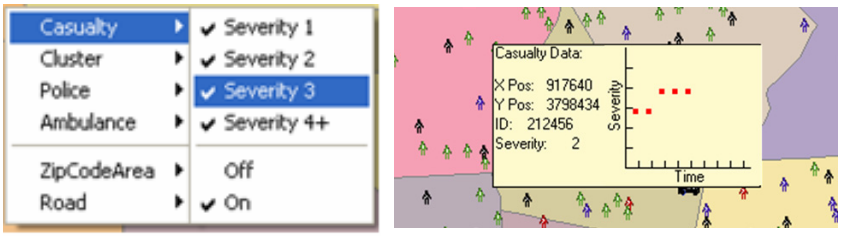

Figure 10. Fast data manipulation and identification: resource context menu (left), and pop-up window for object identification and the visualization of its trend (right).

4.3 Quick Identification

The information specific to each casualty is stored over time and is available to the user by simply hovering over a casualty icon on the screen. A pop-up window then appears which shows the pertinent information (Figure 10, right). Since it would be impossible to track each casualty if thousands were present the information regarding casualty clusters will be more useful for time aggregation analysis (section 4.4). Therefore, specific areas can be monitored during the simulation.

All ambulance route reports in the simulation come with road links Id's, which define the routes for the ambulances. It is possible to store all road link Id's in a report database and use that Id with a search algorithm to retrieve all related information by accessing a link database. However, this will increase runtime computation, which further lowers the performance. An easy and compatible solution to the problem is to store the object index in a container of road links objects. In this way, a search algorithm only needs to be run once when a report is received. The data can then be retrieved very quickly by just accessing the container by object index. We have used the binary search algorithm to find index of road link object in a large container. This algorithm needs $O(\log N)$ iterations in worst case to find an object in a container, which is acceptable for database of about 30,000 objects. To make efficient use of runtime computer memory it is desired that all the information regarding road link data be stored only once so that it can be retrieved as needed by accessing its unique Id (Fig. 11). Each file from the RTI contains multiple routes at a time which show a separate route for each ambulance. A hierarchical dynamic array structure has been created to store a report time and multiple route information at that time for each report. Each route can then be accessed by road link index in that array database.

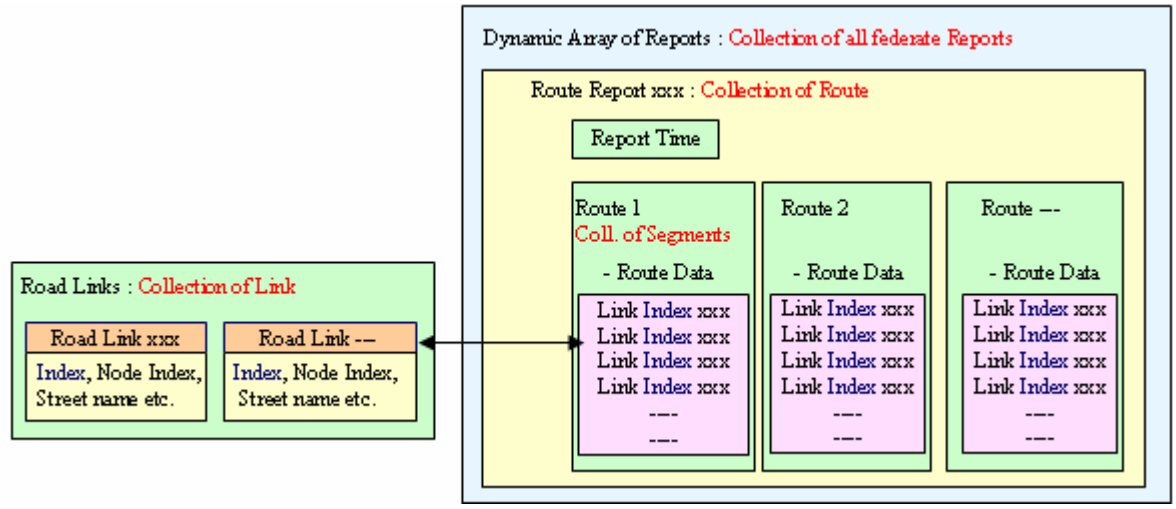

Figure 11. Database design for information storage.

As shown in Fig. 12, ambulance paths can then be highlighted on the screen by merely hovering over them with the mouse. This allows a user to get a clearer understanding of the path and the specific road it may be located on. In order to optimize performance for graphical output, only the current path with the mouse tip is refreshed during mouse hovering. Therefore, the entire display does not have to be refreshed which can distract the user. To catch this mouse event, a small rectangular region is defined around each road link by considering its spatial orientation in the 2D window. As soon as any region catches the mouse tip the searching algorithm will exit with the current route index from the main report index. Based on the current ambulance route index a graphical region is created by joining small oriented rectangular regions of road links which will make the ambulance route. Finally, the system invalidation function is called to refresh only that region in the graphical output window. A backup route index is also stored in run-time memory which is helpful in refreshing the same route region again when the mouse tip moves out of that region.

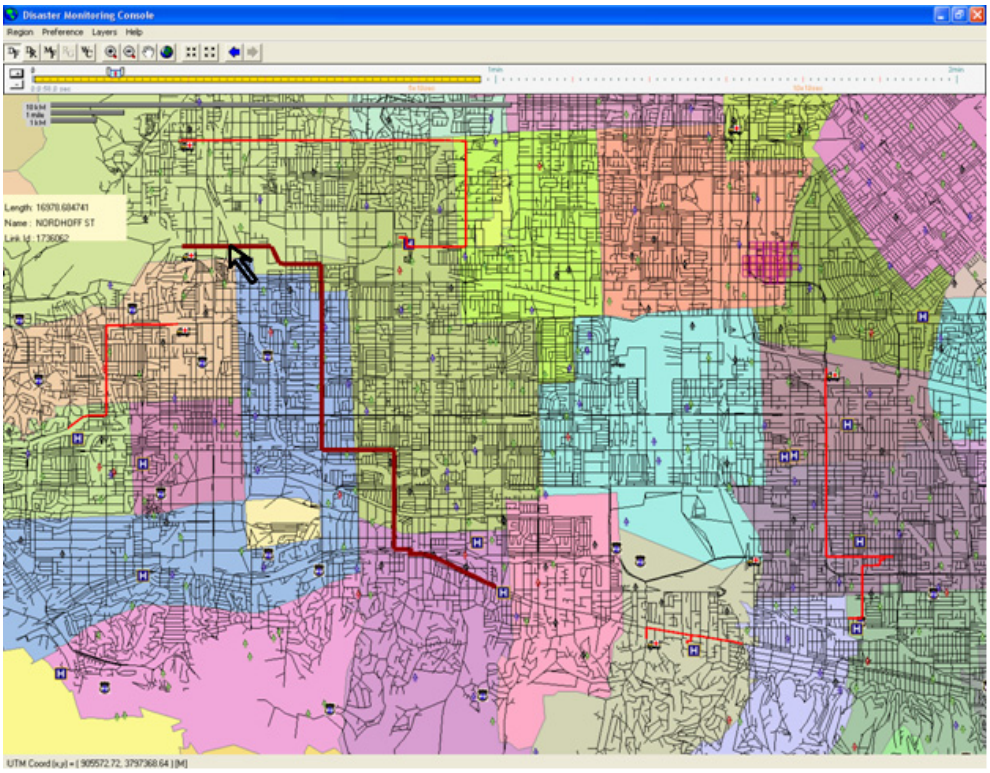

Figure 12. Ambulance route display in an urban area of Northridge region.

4.4 Clusters: Regional Information

A cluster can be defined as the extraction of large amounts of data and displaying it as a group in a vast database system. In a post-disaster simulation the need for dealing with large sets of data and comprehending abstract information is a key to understanding situation awareness. Work on clustering with fused data can be found in applications for a battlefield [21] and the visualization of large datasets [20]. In our simulation, the casualty cluster information is produced by the Level – 2 fusion federate [22].

4.4.1 Formulation of Cluster and Boundary Formation

As an approach to cluster visualization, the entire Northridge area was divided into cells with horizontal and vertical grid data. Fig. 13 (1) shows a cluster with a group of cells which defines the cluster. The outlier information is helpful in the identification of the area the cluster covers, as well as, interpolating intermediate shape information while morphing. Cell information is also effective in the quick identification of areas with the highest emergency within a cluster. In the figure, the opacity of a cell represents the number of casualties present and allows for quick identification of the most troubled spots. It is also possible to see clusters when only considering one level of severity by using the context menu. In addition to the cluster, a boundary can help a viewer recognize the shape of the cluster. With only boundary information, the trend of the regional shape can easily be comprehended (Fig. 13, right).

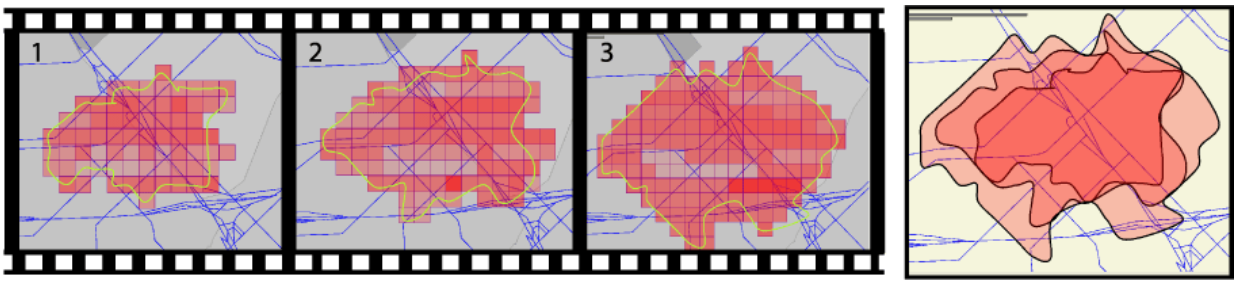

Figure 13. Situation awareness by cluster morphing. The casualty clusters between two discreet time steps (1 & 3) are generated by interpolation (2). The expanding trend of such cluster boundary is shown for better understanding of the situation over time (right).

4.4.2 Dynamic Visualization by Morphing

It is difficult to capture the trend of a situation from one specific time to another. In most cases, the user has to depend on his/her memory to relate the states. In order to provide a way of seeing how the clusters change with respect to time, we incorporated morphing into the simulation. All the visualization elements, such as position, color and shape were stored at each discrete time step. Corresponding discrete states were then interpolated to create the morphing. However, the associated cluster ID may change from one point to the next which makes it difficult to morph the clusters directly. Therefore, morphing was carried out on a cellular level. This way individual cells, which comprise the clusters, may appear and disappear from subsequent states to help inform the user. Using the morphing interface, a user can keep track of a cluster and cells, comprehend the trend, and have effective awareness of the situation (Fig. 13).

5. CONCLUSION AND FUTURE WORK

The main attribute of the present work is to achieve a user-friendly and intuitive visualization methodology for massive post-disaster data fusion in a run-time HLA/RTI network. We developed an integrated GUI framework that provides the capability of fast identification of target objects and manipulation of data layers (low level fusion visualization). It also achieves dynamic visualization of time aggregated data for situation awareness (high level fusion visualization). This has been done through the use of 2D and 3D graphics displays in sync to provide relevant perspectives to the user.

We envision that the framework will be the basis for an actual emergency response center which relies on visualization to help allocate resources in the event of a disaster. For this, a series of performance test will be implemented in the future to verify the usability of the interface. Future plans also include visualization of hazardous material, such as chemical spill and toxic plum, and inclusion of tactical or strategic information for the assessment of future threat caused by a natural or man-made disaster.

ACKNOWLEDGEMENTS

This work was supported by the AFOSR under award F49620-01-1-0371. The author gratefully acknowledge contributions from research assistants Mr. Young-seok Kim, Mr. Pritul Shah and Mr. Mathew Mandiak and valuable input from Mr. James Scandale, Drs. Peter Scott and James Llinas.

REFERENCES

[1] D. Ozisik and N. Kerle , Post-earthquake damage assessment using satellite and airborne data in the case of the 1999 Kocaeli earthquake, Turkey. *Proc. of the XXth ISPRS congress: Geo-imagery bridging continents*, Istanbul, Turkey, 2004, 686-691.

[2] J. Llinas, Information Fusion for Natural and Man-Made Disasters, *Proc. 5th International Conference on Information Fusion*, Annapolis, MD, USA, 2002.

[3] L. A. Treinish, Visual Data Fusion for Applications of High-Resolution Numerical Weather Prediction, *Proc. of IEEE 2000 Visualization Conference*, Salt Lake City, UT, USA, 2000 , 477-480.

[4] K. Severance, P. Brewster, B. Lazos, and D. Keefe, Wind Tunnel Data Fusion and Immersive Visualization: A Case Study, *Proc. of IEEE 2001 Visualization Conference*, San Diego, CA, USA, 2001.

[5] T.-J. Hsieh, F. Kuester, and T. C. Hutchinson, Visualization of Large-Scale Seismic Field Data, *Proc. of 2003 High Performance Computing Symposium*, Orlando, FL, USA, 2003.

[6] G. Srimathveeravalli, N. Subramanian, and T. Kesavadas, A Scenario Generation Tool for DDF Simulation Testbeds, *Proc. of 2004 Winter Simulation Conference*, Washington D.C., 2004.

[7] I. Rauschert, P. Agrawal, S. Fuhrmann, I. Brewer, R. Sharma, G. Cai, A. MacEachren, and H. Wang, Designing a Human-Centered, Multimodal GIS Interface to Support Emergency Management, *Proc. of ACM International Symposium on Advances in Geographic Information Systems*, McLean, VA, USA, 2002.

[8] S. Jain and C. McLean, A Framework for Modeling and Simulation for Emergency Response, *Proc. of 2003 Winter Simulation Conference*, New Orleans, Louisiana, USA, 2003.

[9] Y. Kim and T. Kesavadas, Automated Dynamic Symbology for Visualization of High Level Fusion, *Proc. of 7th International Conference on Information Fusion*, Stockholm, Sweden, 2004, 944-950.

[10] Northridge Earthquake, Southern California Earthquake Center. Retrieved on March 20, 2005 from http://www.data.scec.org/chrono_index/northreq.html.

[11] HAZUS, Multihazard Loss Estimation Methodology, FEMA. Retrieved on January 11, 2005 from http://www.fema.gov/hazus/.

[12] P. D. Scott and G. L. Rogova, Crisis Management in a Data Fusion Synthetic Task Environment, *Proc. of 7th International Conference on Information Fusion*, Stockholm, Sweden, 2004, 330-337.

[13] Q. Gong, A. Jotshi, and R. Batta, Dispatching/Routing of Emergency Vehicles in a Disaster Environment using Data Fusion Concepts, *Proc. of 7th International Conference on Information Fusion*, Stockholm, Sweden, 2004, 967-974.

[14] S. Xiaoxia and Z. Quihai, "The Introduction on High Level Architecture (HLA) and Run-Time Infrastructure (RTI)," *Presented at SICE Annual Conference*, Fukui, Japan, 2003.

[15] K. Watson, D. Espinosa, Z. Greenvoss, J. H. Pedersen, C. Nagel, J. D. Reid, M. Reynolds, M. Skinner, and E. White, *Beginning Visual C#.* Indianapolis, IN: Wiley Publishing, Inc., 2003.

[16] USGS, U.S. Geological Survey. Retrieved on January 11, 2005 from http://www.usgs.gov.

[17] TeleAtlas. Retrieved on March 20, 2005 frm http://www.teleatlas.com.

[18] G. M. Karam, Visualization using Timelines, *Proc. of the 1994 ACM SIGSOFT international symposium on Software testing and analysis*, Seattle, Washington, USA, 1994, 125-137.

[19] E. White, C. Garrett, and S. Robinson, *GDI+ Programming: Creating Custom Controls Using C#.* Birmingham, UK: Wrox Press Ltd., 2002.

[20] A. Schilling and A. Zipf, Generation of VRML city models for focus based tour animations: integration, modeling and presentation of heterogeneous geo-data sources, *Proc. of the eighth international conference on 3D Web technology*, Saint Malo, France, 2003.

[21] C. M. Hoffman, Y. J. Kim, R. P. Winkler, J. D. Walrath, and P. J. Emmerman, Visualization for Situation Awareness, *Proc. of the 1998 workshop on New paradigms in information visualization and manipulation*, Washington D.C., United States, 1998.

[22] K. Chen and L. Liu, ClusterMap: Labeling Clusters in Large Datasets via Visualization, *Proc. of ACM Conference on Information and Knowledge Management*, Washington D.C., USA, 2004, 285-293.