Info Vis

Article

Information Visualization
2015, Vol. 14(3) 204–222

DOI: 10.1177/1473871613516292
ivi.sagepub.com
SAGE

Interactive analysis and visualization of situationally aware building evacuations

Jack Guest, Todd Eaglin, Kalpathi Subramanian and William Ribarsky

Abstract

Evacuation of large urban structures, such as campus buildings, arenas, or stadiums, is of prime interest to emergency responders and planners. Although there is a large body of work on evacuation algorithms and their application, most of these methods are impractical to use in real-world scenarios (nonreal-time, for instance) or have difficulty handling scenarios with dynamically changing conditions. Our overall goal in this work is toward developing computer visualizations and real-time visual analytic tools for evacuations of large groups of buildings, and in the long term, integrate this with the street networks in the surrounding areas. A key aspect of our system is to provide situational awareness and decision support to first responders and emergency planners. In our earlier work, we demonstrated an evacuation system that employed a modified variant of a heuristic-based evacuation algorithm, which (1) facilitated real-time complex user interaction with first responder teams, in response to information received during the emergency; (2) automatically supported visual reporting tools for spatial occupancy, temporal cues, and procedural recommendations; and (3) multi-scale building models, heuristic evacuation models, and unique graph manipulation techniques for producing near real-time situational awareness. The system was tested in collaboration with our campus police and safety personnel, via a tabletop exercise consisting of three different scenarios. In this work, we have redesigned the system to be able to handle larger groups of buildings, in order to move toward a full-campus evacuation system. We demonstrate an evacuation simulation involving 22 buildings in the University of North Carolina, Charlotte campus. Second, the implementation has been redesigned as a WebGL application, facilitating easy dissemination and use by stakeholders.

Keywords

Evacuation, visual analysis, situational awareness, emergency response

Introduction

In any emergency or incident involving large urban structures (arenas, stadiums, college campuses), the safety of the occupants is of paramount importance. Normally, every building has a set of passive safety features (sprinkler systems, fire extinguishers) and evacuation plans or routing maps posted at various points within the building. Current research on studying evacuations has focused on large urban structures or street networks and has been based on mathematical and algorithmic approaches. These methods study the problem of evacuating the occupants from the

Charlotte Visualization Center, Department of Computer Science, The University of North Carolina at Charlotte, Charlotte, NC, USA

Corresponding author:
Kalpathi Subramanian, Charlotte Visualization Center, Department of Computer Science, The University of North Carolina at Charlotte, Charlotte, NC 28269, USA.
Email: krs@uncc.edu

structure in the shortest possible time, also known as the egress time. What is lacking in these methods is the ability to handle real-world scenarios involving large and complex structures that may involve multiple buildings, and more important, the ability to react in real-time to dynamic changes in the scenario, such as blocked stairwells or hallways (due to congestion, smoke), and provide timely recommendations to responders who can mitigate damage, injury, or loss of life. For a commander overseeing the evacuation, the ability to clearly and unambiguously understand the rapidly changing situation is critical and useful for optimal allocation of limited resources and personnel. Closely related is the ability to visualize the indoor building geometry and the current location of the responders, providing a highly intuitive spatial understanding for informed decision making.

In this work, we address some of these challenges, with the primary goal of responding to dynamic events in real-time during an emergency. We begin with an existing heuristic-based route planning algorithm,[1] adapt this algorithm within the spatial and capacity constraints for large buildings, and embed it as part of a visual analytic system that permits complex real-time interactions during the event. This allows dynamic changes in the building accessibility to be incorporated and alternative routing assessed in near real-time. To accomplish this, we employ a level of detail (LOD) graph representation of the underlying urban structures that permits real-time recommendations to be presented to the emergency planners and responders for informed decision making. This is combined with realistic models for building occupancy and traffic flow. Interactive visual analytic tools permit quick exploration of possible impacts of the current situation, and the increased situational awareness for emergency commanders and responders permit more optimal use of scarce resources, for instance, by dispatching responders to areas of need in congested sections of a building, handling casualties, and so on.

We begin with a review of current work on evacuation algorithms, followed by a description of the techniques we use to create our three-dimensional (3D) models and building networks. This is followed by a description of our modified capacity-based route planner, followed by a detailed description of the major features of our interactive visual analytic system and its reporting and recommender functions. We describe the use of this system on typical scenarios using our campus buildings as a test case. We describe our earlier work[2] on evaluating our system and its performance via a tabletop exercise, consisting of three scenarios: gas leak in building, active shooter, and an explosion in an adjacent structure. We next describe our extension of the system to handle a scenario consisting of 22 campus buildings toward accomplishing a campus-level evacuation that will ultimately include foot paths and the surrounding street networks. We also describe the computational challenges in scaling the system to larger groups of buildings, and our web-based implementation, that will facilitate portability and easier dissemination to end users, by exploiting cross-platform advantages.

Previous work

The study of computer-based evacuation modeling has evolved with both mathematical and algorithmic approaches. These approaches are categorized as *macroscopic* and *microscopic*. We review these methods as well as the use of interactive visualization in analyzing evacuations.

Macroscopic models

Macroscopic approaches focus primarily on minimizing egress time. The evacuees are treated as a unit or a group of units and moved from source to destination. Interaction among these units is defined by capacity/congestion rules. Linear programming methods based on network flow were one of the earliest approaches that yielded optimal solutions, but at high algorithmic cost, making them impractical to use in real-world scenarios; for example, the solution to the maximum flow problem has been implemented, with costs as high as $O(n^3)$[1] yielding a best-known cost of $O(nm \lg n^2/m)$. Kamiyama et al.[3] apply two initial conditions to the network flow problem. For each vertex, the sum of transit times of arcs on any path takes the same value, and for each vertex, the minimum cut is determined by the arcs incident to it whose tails are reachable. These assumptions resulted in a two-dimensional (2D) grid network, and they solved the transhipment problem in $O(n \lg n)$ time. Shekhar and Yoo[4] compare models relevant to the study of nearest neighbor paths. Also Kim et al.[5] discuss contraflow in reconfigured networks for emergency route planning. In our work, this is relevant since dynamic changes to the building structure such as blockages or heavy congestion will require modification of the building network, followed by rerouting the affected occupants.

Our model is based on a heuristic approach, the Capacity Constrained Route Planner (CCRP) algorithm proposed by Shekhar et al.[1] This approach attempts to find lower cost algorithmic solutions at the expense of the detail of each evacuee's egress. These approaches are interesting because they can be evaluated quickly from a user perspective as the network flow problem is reduced to a generalized shortest path problem. The inputs are a graph of the building (or

groups of buildings) and the evacuee population. The graph structure consists of nodes comprising various building elements (rooms, corridors, stairwells/elevators) represented as nodes and weighted edges (by distance) representing the relationship between the nodes (paths). The output is a route plan with start times, and a location matrix for each evacuee for each defined time segment. We have adapted this algorithm to meet the more challenging requirements for near real-time decision making in large urban environments, as well as the ability to inject situational changes during an emergency.

Microscopic models

Microscopic approaches use *agent-based modeling*, where each evacuee is governed by unique rules of behavior. Interaction among individuals and their environment are defined based on spatial and social parameters. In addition to the deterministic path or goal, they also rely on behavior rules applied to each evacuee to overcome the "lack of detail" inherent in network flow or heuristic planners. The goal in these methods is also to minimize the time to evacuate individuals to safe zones. Agent-based methods can be adaptations of neural networks and fuzzy-logic[6] toward building evacuation simulation. Discrete particle swarm optimization,[7] use of velocity, and spatially based rules of interaction[8] are other approaches, as detailed by Castle and Crooks.[9] The most important aspect of agent-based models is the rules applied to each evacuee. Castle and Crooks[9] describe a detailed list of rules and attributes. We believe our system captures sufficient detail using a congestion model.

Visual analytics

Visual analytics involves effectively combining interactive visual displays with computational transformation, processing, and filtering of large data.[10] One focus of visual analytics is real-world problems involving situationally aware decision support. Andrienko et al.[11] described the sequence of tasks that are fundamental to exploratory visualization of spatio-temporal data. They classify their work along two dimensions, namely, temporal characteristics (spatial location, existence, etc.) versus state of the data (temporal instant, interval, progression over time, and summary). They incorporate these ideas into effectively demonstrating the movement of storks across a geographical area. Many of these ideas are also part of our system in the effective use of animation, prioritized display of significant or critical events during a simulation, and their interrelationships across different views that make up the final display. Campbell and Weaver[12] investigate situational awareness during emergencies using two different tools: RimSim Response! (RSR) and RimSim Visualization (RSV). Extensive role-playing via serious games and post-mortem analysis is the strategy to continually train responders to more effectively respond during emergencies. The RSR tool supports role-playing games on simulated emergency scenarios, while the RSV tool provides interactive visualization tools that support exploration in a nonsequential manner. The authors present results of a hospital evacuation simulation, which involved evacuating 200–250 patients to 20 other hospitals participating in the tabletop and live drills of the exercise. While their tools appear primarily for planning, the long-term goal of our evacuation modules is for use during an emergency; the effectiveness of the use does depend on live information coming into the tool (our system can dynamically adjust to new conditions). We discuss these further in the concluding sections of this article. The work of Kim et al.[13,14] focused on the use of mobile devices for situationally aware emergency response and training, and thus, their approach is similar to our work. They demonstrated their system with an evacuation simulation of the Rhode Island club fire of 2003. Our system is considerably more general and scalable to large urban buildings and has the means to interrupt the simulation based on new situational information or dynamic changes.

From a visualization standpoint, the use of linked views is important in connecting different representations of information within a single visualization display, and applications specific to urban structures have appeared earlier.[15,16] Meiguins and Meiguins[15] report on the use of coordinated views to analyze an evacuation simulation performed with building occupants tracked by radio frequency identification (RFID) tags. This is, however, an unrealistic expectation in real-world evacuation of large urban environments but might be of use in understanding user behavior as part of emergency planning. Ivanov et al.[16] report on experiments monitoring a 3000-m^2 office space of 80 people using an array of cameras and motion sensors over a period of 12 months. While this raises privacy issues, some level of monitoring in office spaces is becoming the norm; the acquired information can be used during emergencies to monitor movement and for estimating the number of occupants in buildings. Finally, Andrienko et al.[17] propose 2D summary views of transportation schedules for use in evacuation planning, taking into account the different categories of endangered people (sick, injured, disabled), their sources, destinations, and transportation modes. These tools are useful for planning but insufficient for use during an actual emergency.

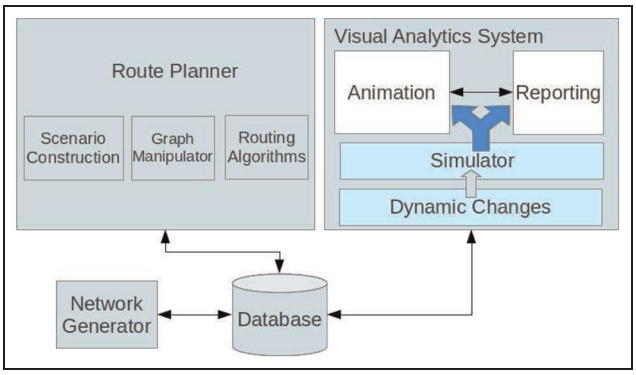

Figure 1. Evacuation system architecture. Route planner involves preprocessing and routing calculations that serve to initialize the system, with processed results stored in a database. The visual analytic system uses visual abstraction and scalable representations that permit real-time interaction, injection of dynamic situational changes, and visual analysis.

Methods

Figure 1 illustrates the major components of our evacuation system for situationally aware evacuations. There are two major components to the system. Functions in the route planner involve a significant amount of a priori processing and initialization. The visual analytic system is a highly interactive system that is user driven and can inject and respond to dynamic changes during evacuations. It also consists of reporting and visual analysis functions that can assist an emergency planner in exploring different scenarios, as to the use and deployment of resources, dispatch responders, assess the effect of rerouting occupants, and so on.

In our earlier work,[18] we described a semiautomatic system that constructed a *building graph*, incorporating key elements of a georeferenced urban structure critical to evacuations, such as hallways, stairways, elevators, and entrances/exits. This building graph generator is used to process urban structures and is stored in a PostgreSQL database. We have successfully processed over 70 buildings on our campus using the graph generator. We next describe the main components of our evacuation system consisting of the route planner and the visual analytic system.

Route planner

The route planner loads evacuation objects, which are combinations of urban structures, pathways, streets, and so on. Evacuations are built as combinations of structural objects and route planner objects and saved in the database. The route planner consists of the following components.

Scenario construction. Scenario construction includes loading single- or multi-building evacuation objects, selecting a route planning algorithm (currently limited to our modified CCRP[1]) and setting visualization modes and evacuation parameters, such as building capacity, egress width, evacuee speed and density, and stairway resistance.

In our scenarios, we use an egress width of 6 ft.[19] Building capacity is estimated as a percent of the maximum classroom occupancy (summed up across all the classrooms in the building and known occupancies in other parts of the building). Classroom occupancies are known based on course schedules and can be used in real-world scenarios. Classroom occupancies are 3 ft^2 per occupant plus 3 ft average separation between seats.[19] Rooms less than 100 ft^2 are assumed to have a single occupant (such as an office). The entire campus evacuation scenario uses actual room capacities taken from the course schedule.

Evacuee speed is set at 3 ft/s,[20] which is reasonable for urban structure building design considering a fire emergency. The maximum density is set at six occupants per 9 ft^2, a bit on the conservative side. In our model, evacuees move only when the density within this space is less than this maximum, else they are stopped until the density falls below this threshold (when some of the occupants in front begin moving forward).

We also include a parameter to induce occupants to choose exits on the current floor, as opposed to possible exits on adjacent floors. This parameter is termed "stairway resistance" and has the effect of increasing the path length up the stairs to the adjacent floor. In our experiments, we set the stairway resistance to 10 times the actual stairway path length.

LOD graph construction/manipulation. All computed egress paths are saved in the database as node to node connections. Nodes are defined based on their function and can be rooms, hallways, stairways, elevators, entrances and exits. The evacuation application also creates muster points, which represent locations where evacuees are ordered to congregate during an emergency. During the process of extracting centerline points, the geometry of hallways is sampled at approximately 2-ft intervals. This process is necessary to ensure accuracy, particularly at corridor elbows. However, this raises computational issues with extremely large buildings or multi-building graphs and hinders real-time performance (due to expensive routing calculations) when dynamic changes need to be accommodated during an event. We address this problem by simplifying the graph with two different levels of detail.

Level 1. Remove redundant nodes. In this step, we simplify the original building graph by removing nodes

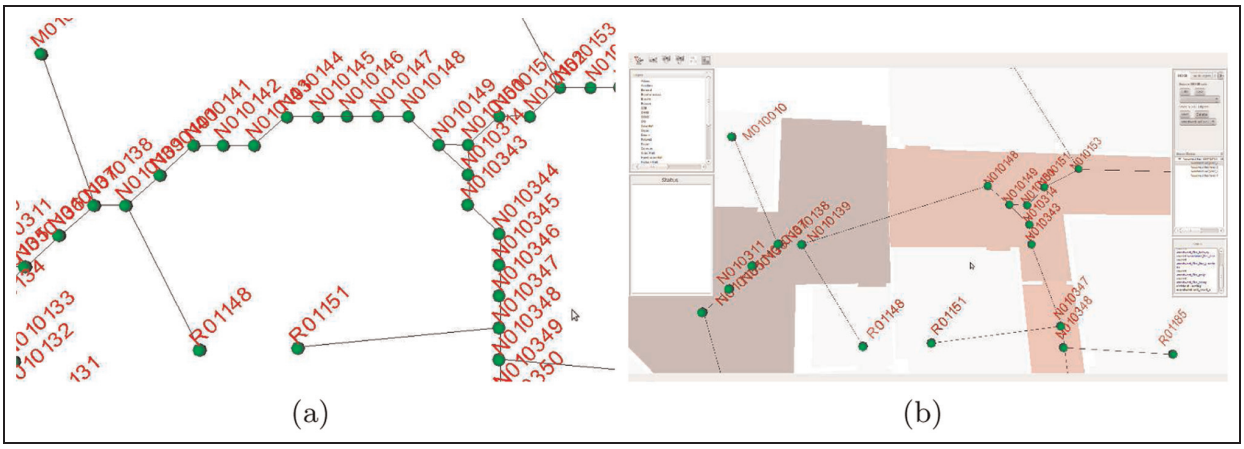

Figure 2. Building graph simplification. Removing redundant nodes: (a) a section of a building floor with labeled building elements and (b) simplified graph.

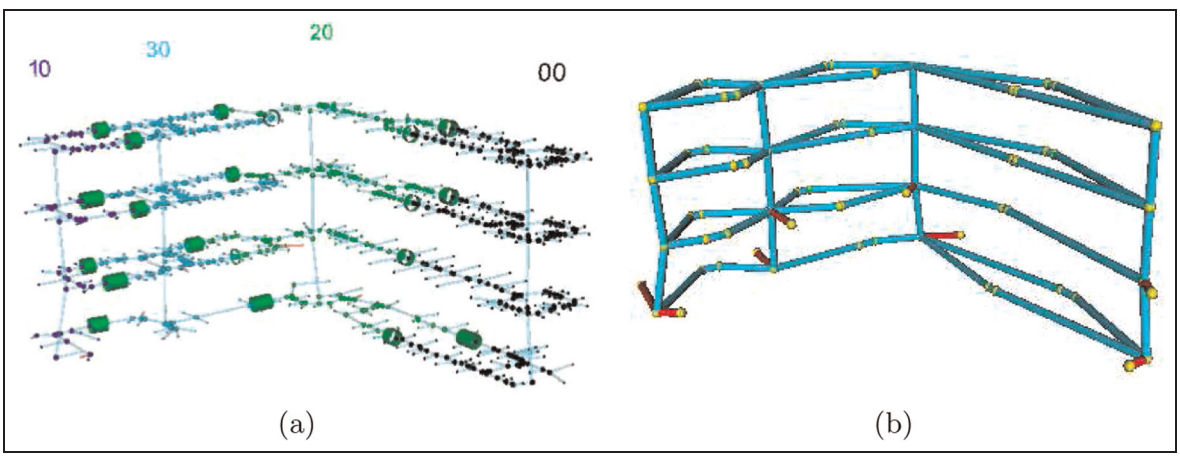

Figure 3. Building graph simplification to a zone graph: (a) building graph of a campus building and (b) transformation to zone graph representation. Yellow spheres are nodes and cyan tubes represent edges. The red tubes represent paths to exits.

that do not impact routing, for instance, shortest path calculations. The larger sampling rate used for accurate centerline calculations results in nodes that can be removed for use in building routes. Nodes and edges are collapsed in the process. Beginning with any node with three or more edges (or any arbitrary node), edges are followed until a node with three or more edges is encountered. This becomes a node of the simplified graph, and a new edge is created connecting to the previous node, as can be seen in Figure 2. The process is continued until all nodes have been visited. Our route planner is executed on these simplified graphs for scenarios in which all original egress paths are available. In our experiments, we obtained a factor of 5–8 reduction in the number of nodes in the simplified graph.

Level 2. Zone graphs. For rapid computation of paths when dynamic changes are injected during an event, the simplified graphs can still be large, especially in multi-building evacuations. In these circumstances, we further simplify the building graphs into *zone graphs*, by segmenting the building into evacuation-sensitive zones: for instance, stairwells, elevators, and exits form the critical elements of any egress path. An example zone graph is illustrated in Figure 3, involving stairwells, hallways, and exits (in red).

To compute the zone graph, we use the precomputed evacuee paths to first associate each node with a zone that is closest to it, using an iterative procedure. In the second pass, the graph connectivity is established by keeping track of the zone of related objects that are encountered in these paths (adjacency lists are

maintained). Intermediate nodes (the yellow spheres in Figure 3(b)) are also identified by paths that cross multiple zones and are further used to complete the graph construction. Zone objects span floors in multistory structures, with appropriate floor identification for proper path determination during routing calculations. The number of nodes in the resulting zone graph depends on the number of zones and the number of floors in the building. In our experiments, a further factor of 5–7 reduction in the number of graph nodes was seen.

Algorithm 1: Modified Capacity Based Route Planner

```
Input: ;
(1) G(N,E): Directed Graph, N nodes, E edges;
(2) Node Properties: capacity, occupancy;
(3) Edge Properties: capacity, travel time;
(4) Set of Source Nodes;
(5) Set of Destination Nodes;
(6) Set of evacuee objects;
Result: Evacuation Plan : Routes with schedules of evacuees on each route
foreach evacuee i at each source node s do
    path_found = find shortest path p from s to all destinations    with available capacity;
    if path_found then
        while p < max_capacity do
                                    /* can route evacuee via p */;
            evacuees[i].path = p;
        end
    end
    move all evacuees;
end
while evacuees not at destination nodes do
    move all evacuees;
end
```

Routing algorithms. We have implemented a modified version of the CCRP,[1] which is illustrated in Algorithm 1. Given a directed graph with node and edge capacities, the algorithm repeatedly computes the shortest path for each evacuee with available capacity. If a path is found, then it assigns as many evacuees as possible through that path, that is, until the capacity of any node or edge along the path is exceeded. This is followed by moving all the evacuees at that time step. The process repeats until all evacuees have found paths to exit the structure. The final step is to evacuate the remaining evacuees in the building (who already have paths but not exited the building).

We have augmented the CCRP algorithm by specifying the movement of the evacuees (in addition to finding the paths) at each iteration. As described in section "Scenario construction," we specify spatial constraints (space occupied and capacity) for each occupant and movement restrictions as a result. Additionally, the algorithm is modified to work with our simplified graphs; in the simplified graphs, weights of the collapsed edges are accumulated and assigned to the new simplified edges. Running the routing algorithm on the simplified graphs makes it more scalable to larger urban structures as well as facilitating dynamic changes to the graph that will require rerouting occupants around blockages or other hazards caused by the emergency event. Finally, although each evacuee has a set average speed (3 ft/s), evacuees cannot exceed the set density threshold. Thus, as congestion builds up, evacuee movements are naturally slowed down. Additional data structures are maintained to make these computations efficient.

Visual analytics system

The visual analytic system (Figure 1) consists of a simulator that accepts user input during an emergency, a 3D interactive animated display of the ongoing evacuation, and reporting and analysis modules. All these views accept direct input, and the views are linked to update automatically.

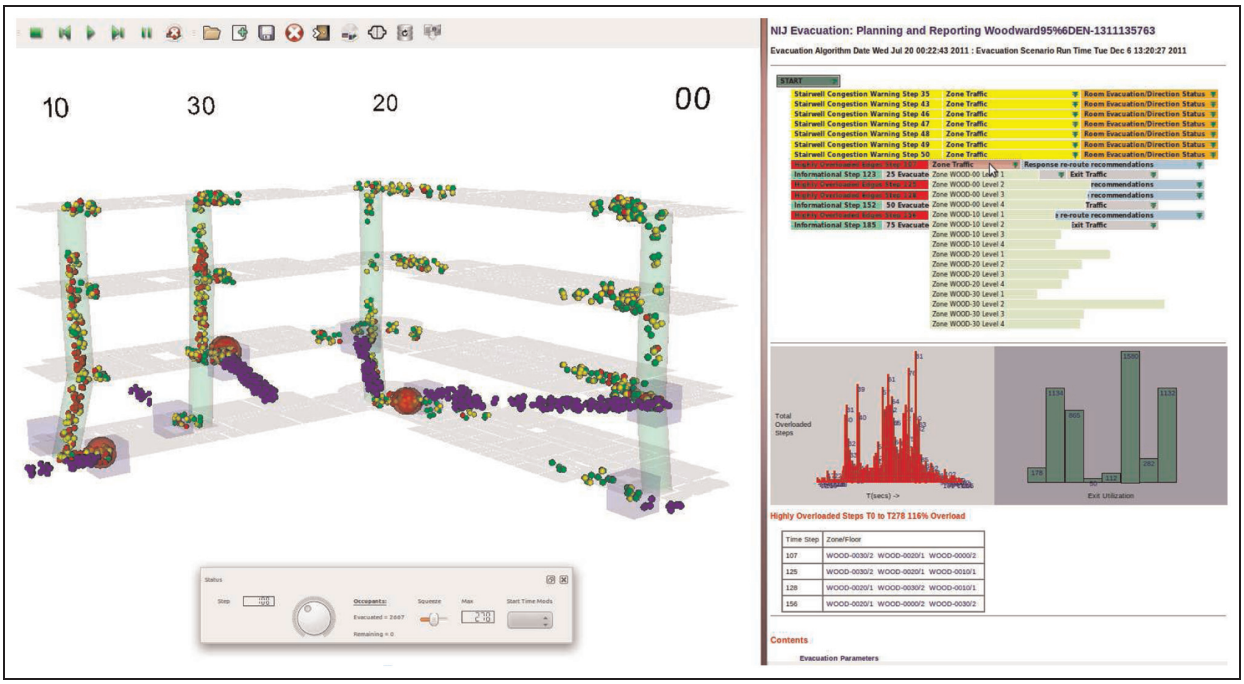

Figure 4. Visualization design. The upper left panel is the 3D view of the building undergoing an evacuation. Spheres encapsulate evacuee population densities, permitting easy identification of congestion points in the building. Vertical tubes (light green) represent stairways/elevators. Cubes on the first and second floors indicate exits. Lower left indicates the status bar for animation control. The upper right panel is for displaying reports of significant events, which can be drilled down. Lower right panels (bar graphs) show aggregate information on exit occupancy as well as events arranged on a timeline. The report views and the 3D views are linked for immediate updates.

Simulator. The simulator loads evacuation plans that were saved for scenarios under normal static conditions. Since dynamic changes affect only a small part of the structure (e.g. a blockage in a stairwell is usually localized, requiring rerouting only occupants close to that area), a large amount of preprocessed data can be reused, contributing to our near real-time performance. The simulator accepts user-defined dynamic changes (specification of blockages or casualty reports through the 3D display or the report modules) and modifies computed paths, and reroutes occupants. In addition, it updates the generated reports and responder recommendations or action plans.

Visualization design

Single-building view. Figure 4 presents our interactive visualization system. Here, we see the occupants evacuating from a single four-story building. Almost all of the interactions are via *direct manipulation.* There are three components that make up the design. On the left is a 3D animated view of the urban structure, where the user can load and play evacuation simulations. Evacuees are represented as spheres and colored green, yellow, or red based on low to high congestion. Partially transparent light green tubes represent stairwells, while purple cubes are exits. Blue polygons represent areas that can be occupied. Large red spheres of varying opacity represent edge (capacity) congestion. On the bottom left is the *status tool widget*, which allows moving around in the animation with a slider, indicating current step, evacuee counts, and total simulation time. The top right panel is the *significant event* window. The rectangular bars are menus with varying levels of detail of the simulation report. The bottom right panel is a scrolling widget with interactive charts and graphs for interacting with the simulation and visual analysis.

Major features within the single-building view are as follows:

- *Congestion representation.* Congestion is a primary concern in evacuation scenarios and predictions of future congestion, and mitigation is important to emergency response teams. In our system, congestion levels range from green to red (low to high).
- *Temporal cues.* The color and size of spheres are modified at significant event times. For example,

sphere size is enlarged when the simulator starts moving evacuees from a source location. The resultant pulsing in the animation informs the user of new sources of traffic and likely areas of future congestion.

- *Details on demand*. The *significant event* window in the reporting tool uses a colorized layered menu so that the visualization of significant information is presented as needed by the user. Significant events are evacuation events of interest to first responders in an evaluation of a given scenario. These event definitions came from discussions with University of North Carolina (UNC) Charlotte police. They include the following:

  - *Informational events*. Limited in this implementation to a percent of the occupants that have been evacuated. We currently use 25%, 50%, and 75%.
  - *Warnings*. These are capacity-related events, which are currently set at 90% of calculated zone capacities. They are displayed for every zone at the threshold at any given time step.
  - *Highly overloaded*. These are also capacity-related events, which are currently set at 116% of calculated zone capacity. They are displayed for every zone at the threshold at any given time step. This happens when a zone is completely full because each zone shares a node with its neighbor zones. This causes the zone density to rise above 100% because it includes the occupants of its shared node.

These events allow a maximum amount of reporting while allowing for quick event scanning in the event report. A user sees a limited top-level distribution of data unless there is a reason to drill down deeper into the event. In the top right of Figure 4, the user has selected a *Heavy Zone Congestion* item (in red) with its time step. The user can explore further via a mouse-over operation to reveal a bar graph that shows the relative congestion of each zone in the building.

- *Interaction on linked views*. The simulation is manipulated by direct interaction over the 3D animation and report views. For instance, a blockage can be introduced via the 3D view, and simulation rerun to generate new (rerouted) paths for impacted evacuees; the report view is updated to reflect the situational change. Similarly, the interaction with the reports menus, charts, and so on temporally updates the 3D animation view. All such operations are performed in near real-time, as the computation is performed on simplified graphs.

Campus view. Figure 5 illustrates an evacuation simulation of our campus; in this simulation, 22 academic, administrative, and student resident buildings were involved in the evacuation. Rather than show the evacuation of each building in its entirety, we show a more abstract view of the evacuation so that the incident commander can get a quick overview. We thus use a 2.5D style visualization in the campus view. The building outlines have been extruded to show a 3D view of the campus; the campus buildings are viewed from above, and bars representing the total occupants in that part of the building are drawn at each time step. Each bar represents a 100×100-ft^2 area, and occupants within that area are summed up across all the floors. In order to reduce clutter, only the areas containing significant numbers of occupants are displayed (using a predefined occupancy threshold). As congestion areas tend to be concentrated near stairwells (which form parts of exit routes), this strategy is reasonable; bars are colored relative to the size of the building occupancy, while the size of the bars are based on the total occupancy across the entire campus. Thus, it is possible to quickly understand the peak occupancies in each building as well as finding peak congested areas in the campus; occupancy ranges are mapped to a discrete set of colors, ranging from green (low) to red (high). Finally, selected areas represented by green cylinders are designated as staging areas for occupants as they exit the building. As the evacuation progresses, these cylinders grow taller, as a function of the number of occupants. Interactive querying for details of a part of the building or the staging areas will be supported in the final implementation, permitting the incident commander to obtain quantitative information during the emergency.

Thus, the campus view provides an overview of the evacuation for a large group of buildings; users can then select a particular building to bring up the building view (see Figure 4). Alternately, if the focus of the evacuation involves a few specific buildings, then these can be selected, and the detailed evacuation of the involved buildings can be analyzed. It is to be noted that the evacuation of all the buildings in the campus view runs concurrently, regardless of what buildings are selected in the current view; the building view simply shows a more detailed view of the occupants and associated geometry (corridors, stairwells, elevators, rooms, exits/entrances) at any instant.

Implementation

Unlike our earlier system,[2] our current system uses web-based languages and tools to make it highly portable. All of 3D rendering is done using WebGL,[21] an

Figure 5. Campus view. Frame from a simulation of the evacuation from 22 academic, administrative, and student resident buildings of UNC Charlotte, with about 15,000 occupants. Colored bars above the buildings indicate occupancy counts, each bar representing a 100 × 100-ft^2 area that spans across all the floors of the building. The color coding is normalized against the maximum building occupancy and the height of the bar is normalized against the maximum occupancy of all buildings involved in the evacuation. Large green cylinders indicate staging areas where evacuated occupants gather; the exact staging area chosen depends on the exit used by the occupant to leave the building. This frame shows the activity at 114 s into a 633-s evacuation simulation.

implementation of OpenGL for web browsers. We use the PostgreSQL database with the PostGIS extensions to maintain all information related to the building geometry and for server–client communication. The database is accessed running on the local machine with direct package calls. The reports section is an HTML/JavaScript window inside a QT4 widget. This allows easy porting to mobile device browsers as well as easy dissemination to system users.

Following are the list of major changes that have been implemented as part of the redesign:

1. *Evacuation planning*. Since the evacuation planning using the CCRP algorithm is computationally intensive, we retained our earlier approach of using the desktop application to compute the evacuation plans as a preprocessing step and store the results in the database. As described earlier, dynamic changes (blockages, rerouting) generally impact a small number of buildings, necessitating updates of their precomputed evacuation plans. All of the data (evacuation scenarios) produced in the database are then available for access by the visual analytics subsystem.

Table 1 illustrates the performance of the CCRP algorithm performance as the number of buildings involved in the evacuation is increased. As expected, CCRP execution time is affected primarily by graph size, which impacts the shortest path computation.

Table 1. CCRP algorithm performance as a function of number of buildings involved in the evacuation 1.

No. of buildings	No. of graph nodes	No. of evacuees	CCRP time (min)
1	1026	3200	0.7
4	3650	7600	3.8
22	12,227	15,200	16.3

CCRP: Capacity Constrained Route Planner.

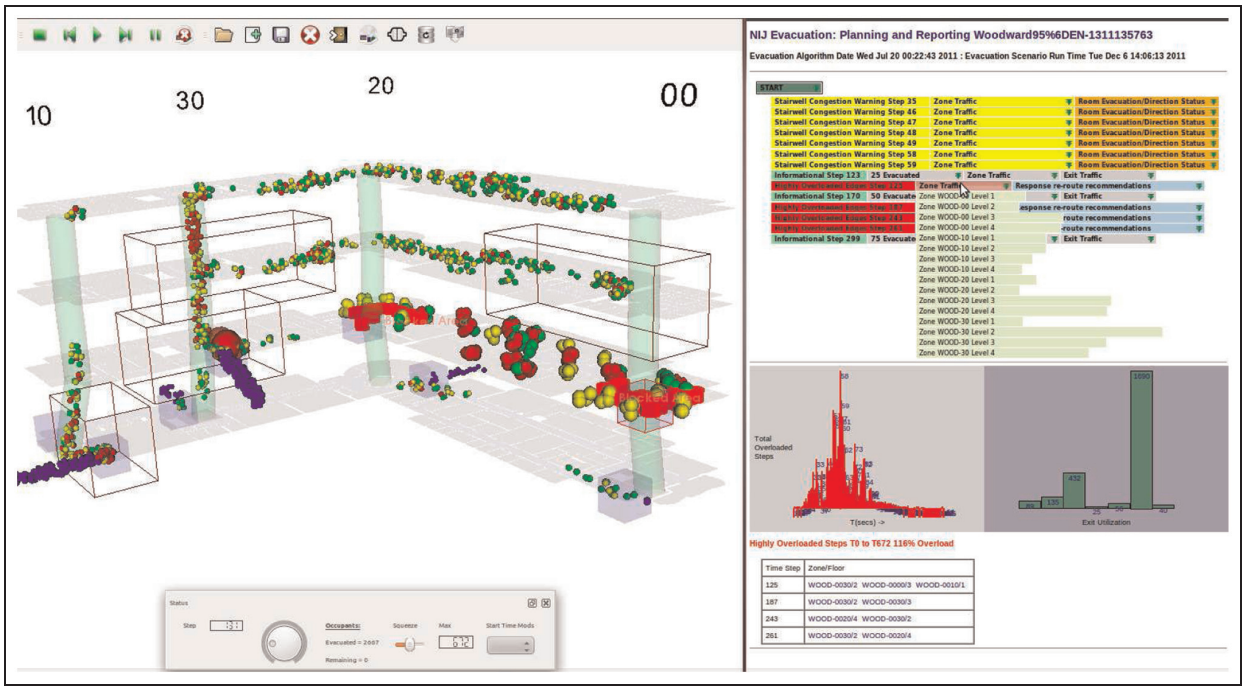

Figure 6. Two blockages have been placed in the building at 45 s into an emergency evacuation. (1) Floor 2 at zone 20 stairwell and (2) floor 2 at zone 00 stairwell. Individuals are shown trapped between them by enlarged spheres. Wireframe cubes indicate traffic flow areas, obtained by rolling over the bar chart at the bottom of the report window.

Recall that the CCRP algorithm performs this computation for each evacuee until all evacuees have a path. The number of exits in a building and the number of larger rooms at long distances are also factors in the CCRP execution time.

2. *Browser visualization*. All of the visualizations are directly supported on the web browser; the animation view is implemented as a WebGL[21] application. WebGL is a browser-based implementation of OpenGL ES 2.0 and uses the GL Shading Language[22] for shaders. The three.js library[23] is used for graphics and rendering. Given that current implementations of WebGL can exploit graphics hardware, very little is lost in terms of geometry rendering efficiency, while application portability is greatly increased.

The WebGL application loads campus building data (via files in "Shape" format); a triangulation is performed, followed by an extrusion to create a 3D model. These building models are then concatenated into a single buffer to improve rendering efficiency. The application follows the same procedure for campus footpath geometry. Animation data consisting of the occupancies centered around each area are loaded, followed by generation of 3D bars scaled and colored by the occupancy at each location. The building geometry is rendered using a lambertian lighting model with backface culling. Bars are drawn with a flat color, and no lighting to reduce shader computation. Screen space ambient occlusion is used to help distinguish the buildings and bar graphs from the background.

The remaining views that include generation of congestion reports and exit statistics are implemented using web technologies (HTML) and remain the same from our earlier work (see Figure 6, right).

3. *User interaction*. All of the interactions involving specification of blockages and rerouting of occupants around inaccessible areas of the building are sent as requests to the server for processing and signaled back to the browser upon completion. This approach avoids significant data processing within the browser. The HTML5 functions related to the canvas for WebGL, websockets for persistent connectivity, and webworkers for true threaded browser operations permit this design and allow for a rich and portable evacuation application.

Example scenarios

Next, we describe four experimental scenarios to illustrate the use of our system. The first three of these

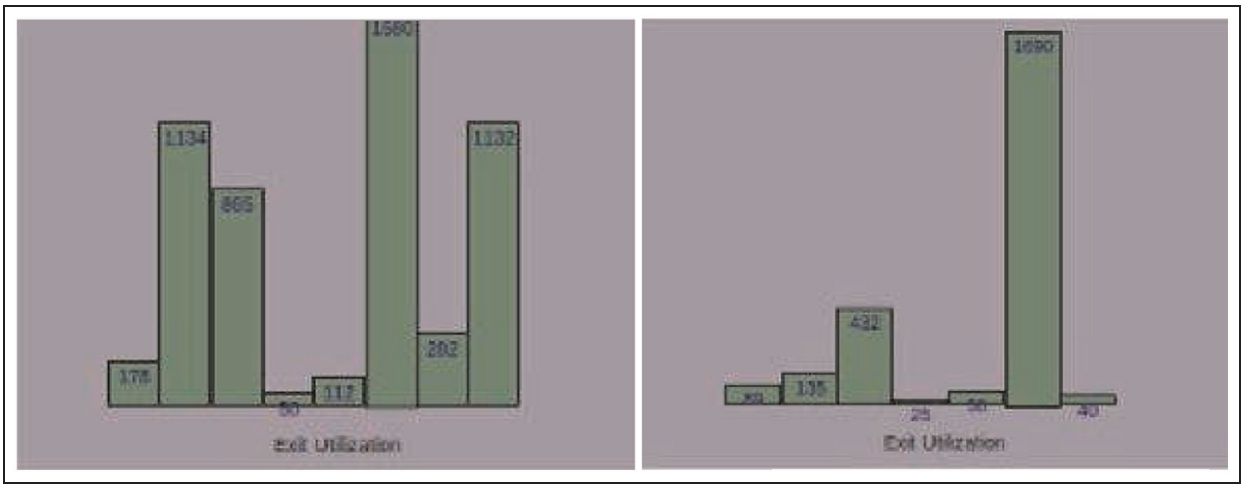

Figure 7. Before and after congestion: exit utilization.

involve a four-building cluster to illustrate the detailed evacuations within a building. In this scenario, there are 3500 evacuees. The maximum egress capacity is set to 1 evacuee per cubic foot and navigation speed is set at 3 ft/s (congestion can slow down or halt evacuees during a simulation). The building is loaded to 95% capacity.

The fourth scenario involves a campus evacuation involving 22 buildings. This scenario involves nearly 15,000 evacuees. It provides a high-level view of the campus evacuation with the ability to select particular buildings to see the details of the location of occupants within the building, congestion areas, and so on.

Situation 1: no blockages

Figure 4 is a screenshot of our visual analytic system, loaded with a campus building with no inaccessible areas. Evacuees are represented by spheres, visually clustered based on egress width. A small red sphere indicates an evacuee cluster on a congested step. Large red spheres of varying opacity indicate congestion greater than 116% of rated capacity. In the time step shown in Figure 4, the user has clicked on the reporting panel (upper right), representing a highly loaded event at time step 108 s. The user has also rolled over the zone congestion bar to reveal the detailed zone congestion graph. As indicated by the bar at zone 30, level 2, this is the most traveled and congested route. The application suggests that a responder be dispatched to this area. As the user rolls over the associated "Response Reroute Recommendations" menu bar in the list, the recommended action can be made visible.

Access for responders can be found by looking at the green exit utilization bars (lower right bar chart of Figure 4) and choosing a low utilization exit. The zone exits in this scenario that are not utilized are on the first level. This type of information can be a powerful dispatch tool for the emergency commander to make an informed decision.

Each bullet in the significant event list (upper right panel in Figure 4) serves as a visual clue to the overall execution of the scenario. The list covers the entire evacuation. The yellow stairway warnings can be drilled deeper to see which areas are becoming congested. These cues are important for responders to quickly react during the beginning of an event or if a campus lockdown has been released. The room evacuation and direction status options can be rolled over to indicate the direction from which the traffic is proceeding, which in turn could result in congestion at a later point.

Situation 2: induced blockages

As shown in Figure 6, two blockages have been introduced into the scenario of Figure 4. The blocked areas represented by red squares spread out over several square feet on the second floor at zone 00 and zone 20. In this example, a total of 3100 evacuees were rerouted and all reporting recalculated in 2.8 s. Note that in the control bar at the bottom of the 3D animation view, the maximum evacuation time has increased to nearly 7 min from less than 3 min.

Figure 7 shows the drastic shifts in the movement of people from a standard evacuation of the building. This example serves to show that evacuation modeling of normal (nonblocked) scenarios is considerably different than when blockages are introduced. In particular, note the difference in the utilization of the exits. In the blocked case, the congestion occurs earlier and is

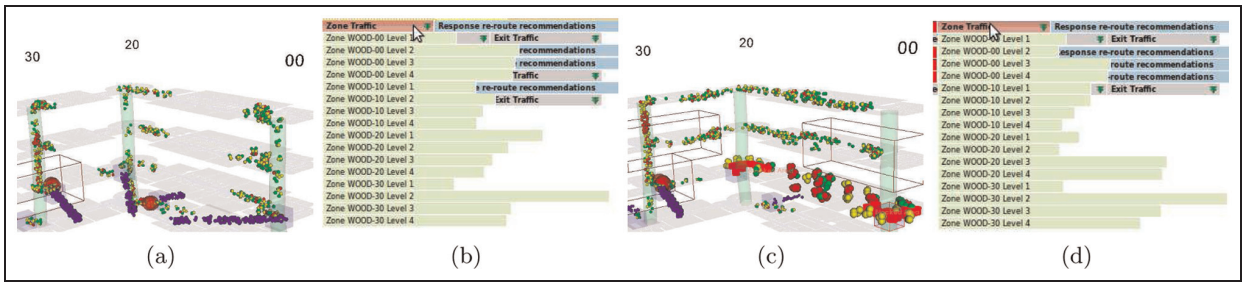

Figure 8. Contrast between normal (unblocked) versus blocked scenarios. Two blockages have been introduced: (a and b) 3D view at 108 s into the evacuation, with top floors mostly evacuated and (c and d) 3D view at 139 s. Floors 3 and 4 are still heavily occupied. Zone traffic (right panels) confirms and illustrates the aggregate picture of the traffic across the entire evacuation.
3D: three dimensional.

steeper, resulting in longer times for all evacuees to exit the building.

Mouse rollover on the significant event list view in Figure 6 shows that the third and fourth floors are getting backed up above the exit at zone 20 floor 2, which is loaded heavily even during a normal scenario. Because the event occurred early, there were a number of evacuees occupying the upper floors.

Figure 8 further contrasts the blocked and non-blocked cases. Here, Figure 8(a) and (b) illustrates the unblocked case at 108 s into the evacuation, and Figure 8(c) and (d) for the blocked case at 131 s. We compare the zone traffic via the light green bar charts. The bar charts show total zone traffic from the beginning to the end of the evacuation scenario. Even though the bar chart for the blocked case is the total picture, it is still clear from both the building and zone traffic charts that the traffic on the top two floors is heavier in the blocked case and, in particular, is shown by the size of the bars labeled Zone WOOD-20 Level 3, Zone WOOD-20 Level 4, Zone WOOD-30 Level 3, and Zone WOOD-30 Level 4. The bar chart and the single-step picture of the building point the user to the same conclusion.

Situation 3: rerouting evacuees

When certain parts of a building are blocked, we can reroute evacuees in that area to other nearby less utilized exits. Also, our system permits a selected number of evacuees to be rerouted to reduce congestion at a stairwell or exit. This operation is performed in near real-time (less than 30 s) in our experiments; however, it depends on the number of evacuees being rerouted. The simulation can then be played to evaluate the traffic or congestion patterns resulting from such an intervention. If needed, a responder can then be dispatched to the affected area for assistance.

Figure 9 illustrates an example evacuation from a cluster of four academic buildings. The original evacuee densities are illustrated in panel 2. Exits 8 and 16 are heavily used, as indicated by the area of their exit circles. The red cubes in panel 1 have been interactively selected (panel 3 shows a zoomed-in view of these areas) for rerouting occupants within those areas. This is followed by specifying the number of evacuees to be rerouted to specific exits (here exits 2 and 23 were chosen). Panel 4 shows the results of these actions, leading to reduced densities at exits 8 and 23.

Building lockdown and release operations can benefit from such "what-if" style scenarios that brings together rich spatio-temporal information into the hands of first responders. In this example, running the entire scenario from initiation to results and analysis took approximately 2 min. When large collections of buildings are involved with traffic routed to the adjacent street networks, such tools can be invaluable for effective and timely evacuation as well as optimal asset deployment.

Situation 4: campus evacuation

As described in section "Visualization design," we have extended the evacuation system from our earlier work[2] to handle evacuation simultaneously from a large number of buildings. Using this campus-level view, we have run a simulation involving roughly 15,000 evacuees distributed across 22 academic, administration, and student resident buildings of the UNC Charlotte campus. Table 2 illustrates the occupancy of these buildings used in the simulation. The evacuation was run over 709 s (time steps), about 11.8 min.

Figure 10 illustrates three key frames during the simulation. The simulation consists of 633 time steps (seconds). In the top frame, at 114 s, peak activity can

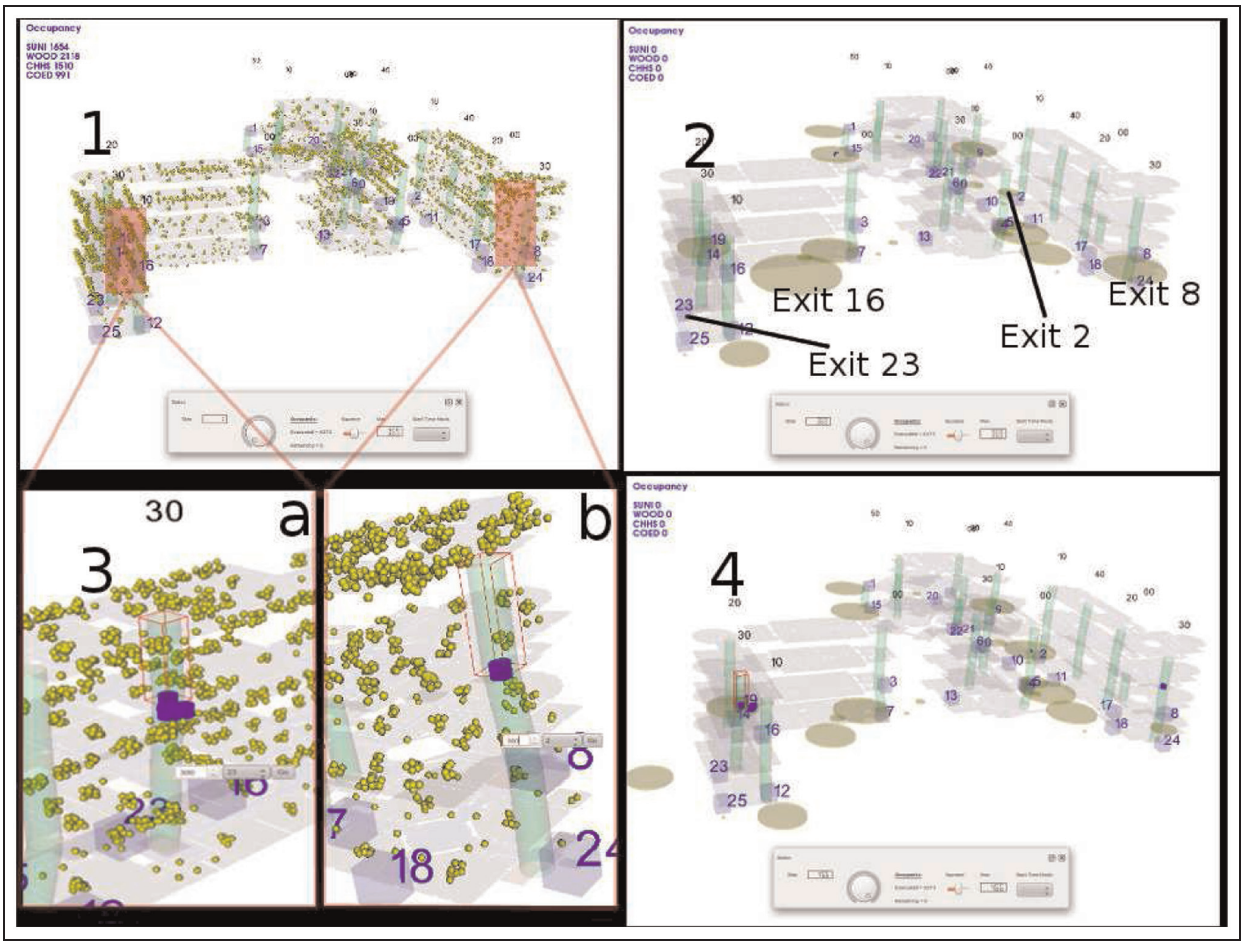

Figure 9. Rerouting evacuees from congested areas in a four-building evacuation simulation: (1) four-building cluster, (2) resulting evacuee density circles after simulation, (3a and 3b) user-added rerouting flags as indicated by the blue disks and associated with the opaque red rectangles in (1), and (4) resulting evacuee density circles after modified simulation, changing the routes of evacuees to exit 2 and exit 23 in their respective buildings.

Table 2. Campus evacuation simulation: involved buildings and their occupancies. There were close to a total of 15,000 occupants on this evacuation simulation.

Building	No. of occupants	Building	No. of occupants
Atkins	557	Grigg	483
Biol	549	Kennedy	201
Burson	471	King	56
Cato	171	Macy	117
Cedar	105	McEniry	876
Denny	123	Reese	539
Coed	837	Robinson	847
CHHS	913	Winningham	61
CRI	778	Student Union	1364
Fretwell	3245	Woodward	1748
Friday	632		

be seen in several buildings, with the red bars indicating high occupancy in each building, relative to its maximum occupancy. The five green cylinders indicate evacuees making their way out of the buildings toward their nearest staging areas. In the middle frame, at 291 s, the evacuation is tapering off, as seen by the large green cylinders indicating the number of evacuees that have exited the building. At this stage, there are 6–7 buildings, which started with a much large number of occupants, that remain to be evacuated. In the bottom frame, at time step 450, evacuation is almost complete, with perhaps 2–3 buildings in their last stages of evacuation.

Evaluation: tabletop exercise

The development of our application has included regular feedback and demonstrations with campus emergency and safety personnel, including the chief of police, other safety officers, and campus business continuity staff. As part of evaluating the system, we

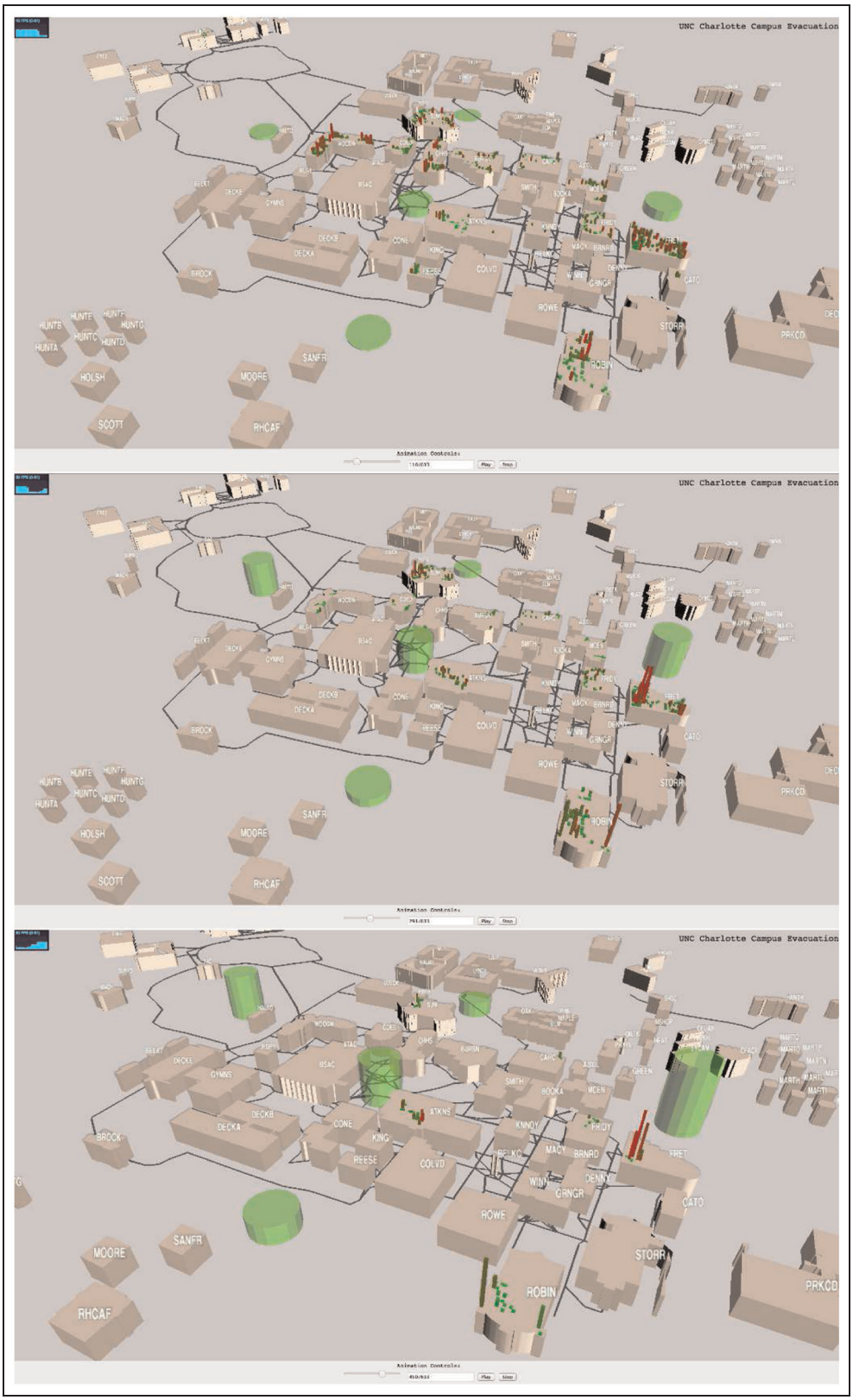

Figure 10. UNC Charlotte campus evacuation simulation. A group of 22 academic, administrative, and student resident buildings were involved in this evacuation simulation, comprising a total of about 15,000 evacuees. Color-coded bars above buildings indicate the number of occupants in that part of the building. Large green cylinders indicate staging areas (muster points) for evacuees to gather for further instructions: (top) evacuation reaches a peak across most of the buildings, (middle) evacuation is tapering off, and (bottom) evacuation is almost complete except for a few buildings with higher populations (note the green cylinders have reached their maximum size).

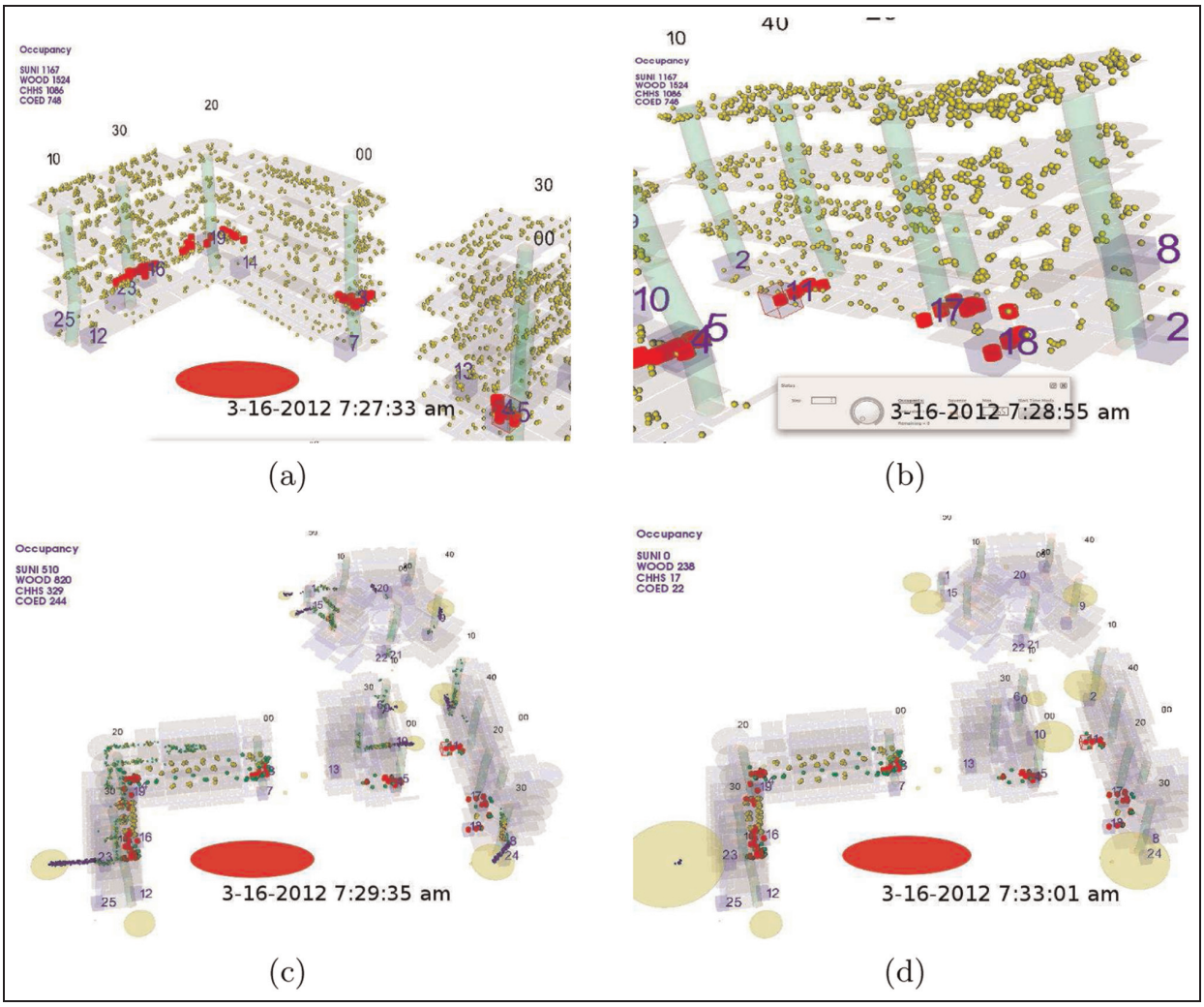

Figure 11. Tabletop exercise: gas leak in building. At approximately 7:20, a gas leak is reported. Lower quad campus possibly affecting four buildings. Evacuation simulation begins. At simulation time: 7:26:38, leak confirmed near red ellipse. Simulation suspended, assets are deployed to prevent evacuation into the hazard. Views modified to simulate asset activities at building exits. (a) Simulation time: 7:27:33—blocking building exits into quad and (b) simulation time: 7:28:55—blocking building exits from adjacent buildings into quad complete. Signal sent for application to perform situationally aware rerouting, (c) simulation time: 7:29:35—visualization of new simulation, evacuation in progress, simulating responders interaction at exits to affected areas and (d) simulation time: 7:33:01—evacuees are avoiding hazard and exiting to safe zones, evacuee densities are indicated by areas of yellow circles.

conducted a tabletop exercise with our campus police. We ran the application through three different scenarios to determine our system's usability, effectiveness, and need for improvements. A business continuity office staff member designed the scenarios. The campus police chief, a senior police officer, and the software team participated in the exercise.

All three scenarios involved a cluster of four campus buildings and a base scenario for the evacuation of approximately 5000 evacuees. The preprocessing step (performed once, at the beginning) was timed at approximately 8 min. All simulations used this base evacuation object. Video of each of the three exercises were recorded for analysis, followed by feedback from the emergency personnel. The system was operated by a member of the software team while commands were received from the police chief.

Scenario 1. Gas leak in building

Figure 11 shows time-sequenced snapshots of a simulated gas leak somewhere in the exercise area. Initially, the gas leak was reported as "near Woodward Hall." The police chief requested a simulation start. As seen in Figure 11(a), the buildings are being evacuated as expected with all exits being utilized. Several seconds

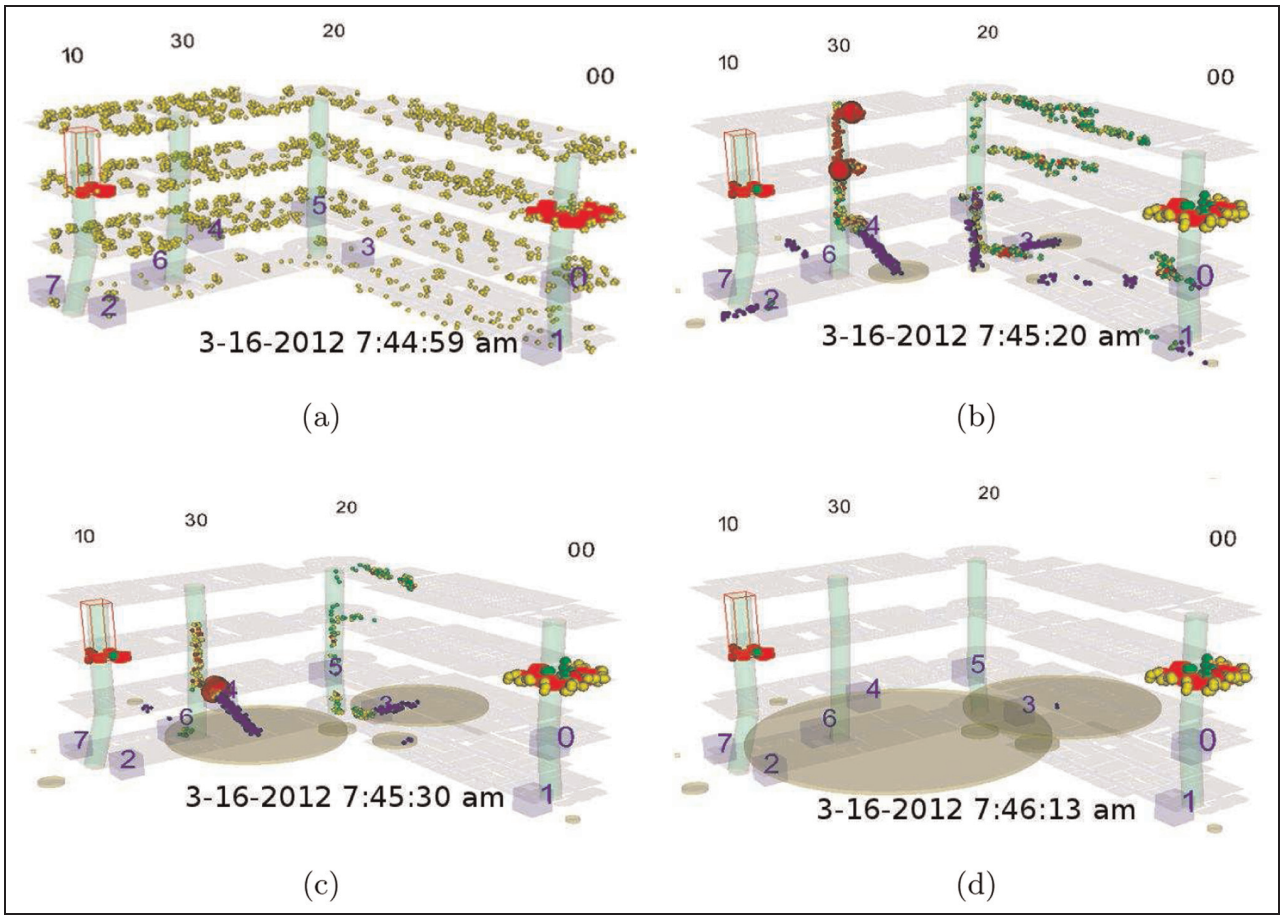

Figure 12. Tabletop exercise: active shooter. At approximately 7:40, a shooter is reported at Woodward Hall. Campus is locked down. Reports are received that third-floor stairwells are blocked at each end of the building. At 7:44:10, blockages are placed in Woodward by the operator, and the simulation is recalculated. (a) Simulation time: 7:44:59—processing for new evacuation simulation is complete, commander orders visualization of new simulation; (b) simulation time: 7:45:20—trapped evacuees noted at third floor zone 00 and zone 10; (c) simulation time: 7:45:30—extreme congestion noted at stairwells in floors 2, 3, and 4 at zone 30; and (d) simulation time: 7:46:13—simulated evacuation complete. Evacuee populations are indicated by approximate area occupied circles.

into the simulation (simulation time: 7:26:38), a report is received that the leak is in the "courtyard," as shown in the red ellipse in the figure. The simulation is halted and reset. The police chief instructed first responders to be dispatched to the building exits facing the courtyard. Also, entrance/exits into the courtyard were to be blocked from further use.

The simulation was restarted based on the new situation. We interacted with the software by placing blockages at the requested areas from 7:27:27 until 7:28:19 (Figure 11(a) and (b)). At this point, the software began to recalculate the 5000 evacuee paths. At 7:29:08, calculations were completed and the reporting process rebuilt, including the scenario timeline and the temporal congestion and exit utilization charts. The police chief requested to see the simulation based on the new situation.

Evacuees are confirmed to be exiting the buildings away from the hazard, as seen in Figure 11(c) and (d). The simulated time to exit all buildings increased from 318 to 687 s. There were large evacuee populations in the areas of Woodward Hall opposite the hazard, and it was noted that due to the blockages in the second floor, some of the evacuees were trapped.

Scenario 2. Active shooter in building

Figure 12 shows time-sequenced snapshots of a simulated active shooter exercise in the Woodward Hall. First, the police chief ordered a campus lockdown and the building to be evacuated. At this point, we switched from the base evacuation scenario of the lower quad (building cluster) to a base scenario of Woodward Hall. We could have placed blockages in

the locked-down buildings but chose to open a single-building scenario for the purposes of the exercise.

Some highlights in this exercise include building rerouting and reporting occurs in 49 s (3 s for evacuee rerouting and 46 s for report generation). The total time here is similar to the multi-building evacuation because our base scenario included 3600 evacuees. This simulates a highly overloaded building to exercise the software for testing.

Scenario 3. Explosion in utility plant

An explosion in the regional utility plant (RUP) building created a scenario where the four-building evacuation simulation of Figure 11 was also used. This scenario also found evacuees blocked in the upper floors, and the explosion created a hazard in the building courtyard. As reports were received, the building floors were blocked, and the simulation was started. As more reports were received, it became obvious that the personnel would exit toward the hazard in the courtyard. The exit density circles alerted the police chief to this problem, and emergency personnel were dispatched to redirect these evacuees. At this point, the police chief requested the exits facing the courtyard to be blocked. The simulation was restarted, and evacuation times and exit results were evaluated as in previous scenarios.

Analysis and system assessment

We detail below both the observations from first responders as well as the important features and current limitations of our evacuation system, as noted from the tabletop exercise. Overall, the feedback from the chief of police (who played the role of incident commander) and his officers was positive and consisted of the following observations:

- The ability to see the 3D layout of the buildings and surrounding areas and get a sense of the current situation was considered the most valuable. The ability to see the evacuation unfold and the buildup at congestion points, and the ability to direct evacuees away from a hazard were considered critically important.
- The near real-time responsiveness of the system and the ability to see the evacuation under blockages were valuable for assessment and taking appropriate action, such as dispatching first responders.
- In the gas leak exercise, a review of the evacuation helped the commander quickly size up the situation (number of evacuees, exit routes, etc.) and order a building evacuation. Once the hazard was located, evacuees were routed away from it by injecting suitable blockages at key points in the building.
- In the active shooter scenario, the police chief noted that the exit utilization and congestion reports would be an invaluable tool for first responders to analyze the condition of a building and dispatch personnel.
- Additional work on the user interface will be needed to further minimize delays during a dynamically changing situation; for instance, blockages are specified one at a time; a "lasso" style interface to specify multiple blockages was considered more intuitive and efficient.
- A limitation of the current system is its inability to localize blockages to the stairways or exits, trapping evacuees in the vicinity. This will be addressed by rerouting the evacuees in the blocked areas to other exits.
- A visualization issue that is common to visual analytic systems is visual clutter and the ability to unambiguously visualize critical information. As we extend our system to incorporate tens of buildings in evacuation scenario, these issues will require careful design and representation choices, with input from first responders.

Conclusion

In this work, we have presented an interactive visual analytic system for situationally aware evacuations of large urban structures. The goals of this work were to provide interactive visual analytic tools that can be used in real-world scenarios (large urban structures, dense collections of buildings) and, more importantly, to be able to run evacuation scenarios in the context of dynamically changing conditions. This is the most important contribution of this work and distinguishes it from previous work on evacuation planning/simulation. We have developed and used an LOD representation of building graphs that can be used as part of a visual analytic system for near real-time response. This in turn permits situational changes to be incorporated into the underlying models and evacuees rerouted. Also, our visual analytic system provides recommendations through the reporting functions that can be used for effective use of scarce resources in dispatching responders to areas of need during the emergency.

We extended our earlier work[2] to handle large networks of buildings by augmenting the building evacuation view with a campus-level visualization, which permits the incident commander to obtain an overview of the evacuation, while permitting any individual building evacuation to be examined in more detail. We

demonstrated our system with an evacuation scenario of 22 academic, administration, and student residence buildings of UNC Charlotte. Depending on the total geometry rendered and the number of occupants, we are able to maintain a near real-time frame rate most of the time (ranging from 4 to 60 frames/s). Future work will require additional simplification of the visualization to handle large neighborhoods, as the system is scaled up to handle urban environments.

We evaluated our system with first responders, including the campus police chief, a senior police officer, and public safety and business continuity/planning personnel. Input from these experienced personnel was invaluable. A tabletop exercise was performed with a smaller cluster of four buildings, with three different scenarios (gas leak, active shooter, and explosion) overseen by the police chief, acting as the situation commander. Overall, the system performed well, as evidenced by direct feedback from the first responders, with valuable suggestions to improve the system.

Extension of the evacuation to the larger campus building network was only recently completed, as well as the transition toward a web-based implementation of the visualization. We are planning formal evaluation of the system with emergency responders. A full-campus evacuation would also require routing occupants via the foot paths to parking lots and out of the campus. This work is only just beginning, as it would require control of the traffic flow, in a manner similar to the contraflow work of Kim et al.[5]

Experiments with our current system takes on the order of about 8 min of preprocessing time, a significant amount of time, given that the evacuation itself might be completed in a fairly short amount of time; the computation time is dependent on the building geometry, number of available exits, and so on. Thus, future work has to focus on the ability to reduce the preprocessing time. There are two ways to significantly reduce the preprocessing time: (1) we can take advantage of known occupancies of urban structures; for campus buildings, it is possible to estimate occupancy based on the knowledge of class schedules, number of offices, and their occupancy (staff, faculty, laboratory). These can be integrated into the evacuation system as part of scenario construction; and (2) a large part of the route planner computation only depends on the building geometry that very seldom undergoes significant change; thus, routes within the building to the exits can be precomputed and stored for later retrieval. Beyond this, improved performance using more powerful computational resources, including more efficient and parallel implementation of key aspects of the route planner will improve the scalability of the system.

Finally, our goal for this evacuation system is for its use during an emergency. While more powerful computational resources are helpful, it is also useful to have knowledge of the number of occupants that remain in the building(s) as the evacuation progresses. In the work of Meiguins and Meiguins,[15] RFID tags were attached to each occupant. This is infeasible in real-world situations and raises privacy issues. Most current buildings (including the newer buildings on the UNC Charlotte campus) include motion and door sensors that can monitor passing traffic. These can be used to estimate the number of occupants entering or leaving a building via each exit, corridors, stairwells, and so on. Such information, if available, can be acquired and integrated into the evacuation system. The accuracy of these estimates can be validated by manual collection of the data over the period of a normal week during a semester (for campus building scenarios, at least) and appropriate corrections applied to the estimates.

Funding

This project was supported by Award No. 2009-SQ-B9-K009, awarded by the National Institute of Justice, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this presentation are those of the authors and do not necessarily reflect those of the Department of Justice.

References

1. Lu Q, George B and Shekhar S. *Capacity constrained routing algorithms for evacuation planning: a summary of results*. vol. 3633. Berlin/Heidelberg: Springer-Verlag, 2005, pp. 291–307.
2. Guest J, Eaglin T, Subramanian K, et al. Visual analysis of situationally aware building evacuations. In: *Proceedings of IS&T/SPIE electronic imaging, 86540G–86540G*, San Francisco, C, USA, 4–6 February 2013, Bellingham WA, USA: SPIE.
3. Kamiyama N, Katoh N and Takizawa A. An efficient algorithm for the evacuation problem in a certain class of networks with uniform path-lengths. *Discrete Appl Math* 2009; 157: 3665–3677.
4. Shekhar S and Yoo JS. Processing in-route nearest neighbor queries: a comparison of alternative approaches. In: *Proceedings of the 11th ACM international symposium on advances in geographic information systems*, New Orleans, LA, USA, 3–8 November 2003. New York: ACM.
5. Kim S, Shekhar S and Min M. Contraflow transportation network reconfiguration for evacuation route planning. *IEEE T Knowl Data En* 2008; 20: 1115–1129.
6. Lo SM, Liu M, Yuen RKK, et al. An artificial neural-network based predictive model for pre-evacuation human response in domestic building fire. *Fire Technol* 2009; 45: 431–449.

7. Fang G. Swarm interaction-based simulation of occupant evacuation. In: *Proceedings of 2008 IEEE Pacific-Asia workshop on computational intelligence and industrial application*, Wuhan, China, 19–20 December 2008. USA: IEEE.
8. Guy SJ, Chhugani J, Kim C, et al. Clearpath: highly parallel collision avoidance for multi-agent simulation. In: *Proceedings of Eurographics/ACM SIGGRAPH symposium on computer animation 2009* (eds E Grinspun and J Hodgins), New Orleans, LA, USA, 1–2 August 2009. New York: ACM.
9. Castle CJE and Crooks AT. *Principles and concepts of agent-based modelling for developing geospatial simulations*. Centre for Advanced Spatial Analysis working paper series 110, 6 September 2006. London: University College London, pp. 1–52. Available at: http://discovery.ucl.ac.uk/3342/
10. Thomas J and Cook K. *Illuminating the path: the research and development Agenda for visual analytics*. IEEE Press, Chicago, 2005.
11. Andrienko N, Andrienko G and Gatalsky P. Towards exploratory visualization of spatio-temporal data. In: *Proceedings of 3rd AGILE conference on geo-graphic information science*, Helsinki, 25–27 May 2000. AGILE.
12. Campbell B and Weaver C. Rimsim response hospital evacuation: improving situation awareness and insight through serious games play and analysis. *Int J Inform Syst Crisis Response Manag* 2011; 3: 1–15.
13. Kim S, Maciejewski R, Ostmo K, et al. Mobile analytics for emergency response and training. *Inform Visual* 2008; 7(1): 77–88.
14. Kim S, Jang Y, Mellema A, et al. Visual analytics on mobile devices for emergency response. In: *Proceedings of IEEE symposium on visual analytics science and technology (VAST)*, Sacramento, CA, 30 October–1 November 2007, pp. 35–42. New York: IEEE.
15. Meiguins B and Meiguins A. Multiple coordinated views supporting visual analytics. In: *Proceedings of the ACM SIGKDD workshop on visual analytics and knowledge discovery: integrating automated analysis with interactive exploration*, Paris, France, 28 June–1 July 2009, pp. 40–45. New York: ACM.
16. Ivanov Y, Wren C, Sorokin A, et al. Visualizing the history of living spaces. *IEEE T Vis Comput Gr* 2007; 13: 1153–1159.
17. Andrienko G, Andrienko N and Bartling U. Interactive visual interfaces for evacuation planning. In: *Proceedings of the working conference on advanced visual interfaces (AVI) 2008*, Napoli, 28–30 May 2008, pp. 472–473. New York: ACM Press.
18. Liu J, Lyons K, Subramanian K, et al. Semi-automated processing and routing within indoor structures for emergency response applications. In: *Proceedings of SPIE conference on defense, security, and sensing*, Orlando, FL, USA, 28 April 2010.
19. Assembly NCG, North Carolina state building code, http://www.ncga.state.nc.us/EnactedLegislation/Statutes/HTML/BySection/Chapter_143/GS_143-138.html (accessed August 2013).
20. CFPA-Europe. European guideline: fire safety engineering concerning evacuation from buildings, http://www.cfpa-e.eu/cfpa-e-guidelines/guidelines-fire-protection/ (2009, accessed August 2013).
21. WebGL: OpenGL ES 2.0 for the Web, http://www.khronos.org/webgl/
22. OpenGL Shading Language, http://www.opengl.org/documentation/glsl
23. three.js, http://threejs.org