Avatar-centric Risk Evaluation

Maria-Cruz Villa-Uriol[†], Falko Kuester[‡]
Calit2 Center of GRAVITY
Department of Electrical Engineering and Computer Science
University of California, Irvine
mvillaur[†],fkuester[‡]@uci.edu

Oscar Garcia Pañella[§], J. Andres Fernandez Munuera[††]
Audiovisual Technologies Department
La Salle School of Engineering, Barcelona, Spain
oscarg[§],andresf[††]@salleurl.edu

Abstract

Hazard detection and prevention of natural and man-made disasters at critical civil infrastructure is becoming increasingly important. Recent events, such as earthquakes and terrorist attacks, clearly demonstrated the societal and economical impact stemming from the limitations and uncertainty within currently deployed emergency response systems. We present a new data visualization and simulation platform that will facilitate risk detection, emergency response and assessment in hazardous situations.

The platform is based on the acquisition, modeling and analysis of sensor data, to capture objects and temporal changes in the observed spaces. Avatars are acquired, inserted and tracked in a virtual environment, enabling the simulation of multiple perilous situations and assisting in determining possible risk mitigation strategies. While the initial research focus is on algorithms and techniques in the field of hazard detection and prevention using path planning, the results can also be applied to diverse fields such as media contents development, cultural heritage, e-learning, teleconferencing, animation and video gaming.

1. Introduction

Although Information Technology (IT) plays no role in triggering natural disasters, it has a prominent role in anticipating, detecting and monitoring them, resulting in minimized loss of life, injuries and resulting damage [7, 8]. Likewise, in our modern society public safety issues are becoming increasingly important for governments, and recent events demonstrate that new approaches to risk analysis, identification and mitigation are urgently needed.

Hazards are generally studied in the context of the environment that they occur in and the direct impacts on humans within these spaces. Therefore it is important to properly model human response while also considering biomechanical constraints.

Badler [2] pioneered the integration of animated avatars in virtual environments, focusing on establishing connections between language instructions and human actions. These concepts have been applied to automatic evaluation of assembly, repair and maintenance tasks [1, 4]. Fire hazards and their environmental effects in buildings [6] were introduced by Bukowski and Séquin [3].

Using virtual environments, we propose to (a) analyze the data acquired by existing sensor networks installed in critical infrastructures, (b) capture and animate avatars to enhance the virtual experience, and (c) recreate and simulate hazardous situations aiming at the design and evaluation of new strategies for emergency management and response.

Figure 1 provides a high-level overview of the system. Given an inventory data set describing the captured environ-

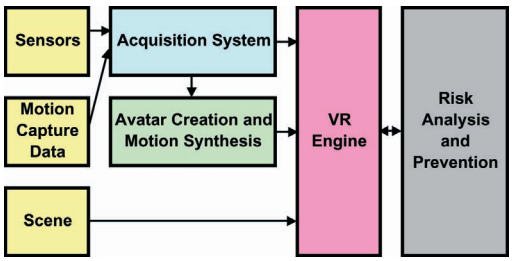
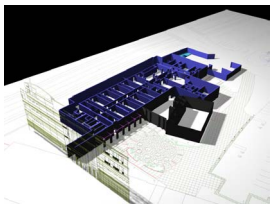


Figure 1. System pipeline.

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

(a) Picture.



(b) 3D Model.

Figure 2. Calit2 case-study building.

ment and using its deployed sensor network, the system is able to continuously acquire (Acquisition System) the current sensor data, (a) enabling an interactive recovery of a subject present in the scene and its motion (Avatar Creation and Motion Synthesis Module), while (b) providing a continuous time-varying sensor data flow needed to drive the Risk Analysis and Prevention Module. This module is in charge of simulating diverse hazardous scenarios (Virtual Reality Engine Module), to establish a risk analysis and mitigation or containment protocols.

The environment used for our simulations is the new building of the California Institute for Telecommunications and Information Technology (Calit2) at the University of California, Irvine (Fig. 2). This institute provides faculty, students, visiting scholars, and industrial partners with an IT-enabled environment to conduct research in communications and information technologies as well as their social implications and it is one of the most heavily instrumented buildings in the nation. Sensors are deployed to monitor ground and building movements, changes in environmental conditions as well as a massive camera network.

2. Acquisition System

Data acquisition is performed at two different levels supporting the recording of static and dynamic content. An optical motion capture system is used for avatar creation and motion synthesis (Sect. 3) required to animate the avatars in the virtualized environment. In addition, the sensor network continuously monitors the environment and provides data for the risk analysis and prevention module (Sect. 5).

Motion Capture Data Using an optical motion capture system, several sequences for different subjects were recorded to provide reference data required to validate the synthesis of avatars and generate new animations used in the visualization stage.

Sensor Data UCI's Calit2 building is instrumented with a sensor network continuously monitoring crucial parameters for our risk analysis and prevention framework, including acceleration, humidity and temperature. Our system can access these sensors remotely, retrieving information including: (a) a unique identifier of the sensor, (b) the latest performed measurement and (c) a timestamp that helps determining if the value is outdated or not. This information is time-varying and the sampling rates are known beforehand. A hashtable is available containing sensor specifications including capabilities and location. For the presented application, we trigger events using scripts. In addition virtual (synthetic) sensors and sensor data can be generated to explore different "what if?" scenarios.

3. Avatar Creation and Motion Synthesis

The system focusses on enhancing the virtual experience when simulating hazardous scenarios aimed at the design of emergency plans that increase the protection level for building occupants. Humans are a key element in these simulations and avatars need to observe proper biomechanical constraints. For this purpose, using optical motion capture data for arbitrary human subjects, our avatar creation and motion synthesis algorithm compactly recovers models and motions in terms of (a) a kinematic skeleton representing the articulated body in a reference pose and (b) a set of motion parameters specified at each instant in time referred to as the kinematic skeleton. These animatable avatar models are finally handed to the Risk Analysis and Prevention module (Sect. 5) to populate the simulated virtual environment.

Five serial kinematic chains approximate our human skeleton (Figure 3). The center of the chest serves as the starting point for these chains: head ($\hat{Q}_{head}$); left and right arms ($\hat{Q}_{arm}$); and left and right legs ($\hat{Q}_{leg}$). We use robotics terminology to describe the joint types and dual quaternions to represent the transformations at each of the joints. These kinematic chains can then be defined in terms of spherical ($\hat{S}(\theta_1, \theta_2, \theta_3)$) and revolute ($\hat{R}(\theta)$) joints. The kinematics equations that characterize each of these chains are

$$\begin{aligned}\hat{Q}_{head} &= \hat{S}_{neck}(\theta_{h1}, \theta_{h2}, \theta_{h3}),\\ \hat{Q}_{arm} &= \hat{S}_{shld}(\theta_{a1}, \theta_{a2}, \theta_{a3})\hat{R}_{elb}(\theta_{a4})\hat{R}_{wrst}(\theta_{a5}),\\ \hat{Q}_{leg} &= \hat{S}_{hip}(\theta_{l1}, \theta_{l2}, \theta_{l3})\hat{R}_{knee}(\theta_{l4})\hat{R}_{ankle}(\theta_{l5}).\end{aligned}$$

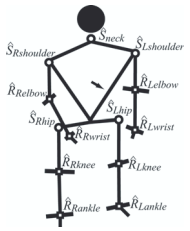


Figure 3. Kinematic skeleton configuration.

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

With dimensional kinematic synthesis it is then possible to compute the kinematic skeleton from a finite number of key poses captured from full body articulated movements of an arbitrary subject. These key poses $\hat{P}^i$ consist of a sequence of coordinate frames aligned to the main body limbs over time, and can be extracted from the available motion capture data. To formulate the synthesis problem, kinematics equations of each chain ($\hat{Q}_{head}$, $\hat{Q}_{arm}$, $\hat{Q}_{leg}$) are equated to the available data ($\hat{P}^i_{head}$, $\hat{P}^i_{arm}$, $\hat{P}^i_{leg}$) at each instant of time t_i, yielding the design equations

$$\mathcal{Q}_{head_i} : \hat{Q}_{head} - \hat{P}^i_{head} = 0, \quad i = 1, \dots n,$$
$$\mathcal{Q}_{arm_i} : \hat{Q}_{arm} - \hat{P}^i_{arm} = 0, \quad i = 1, \dots n,$$
$$\mathcal{Q}_{leg_i} : \hat{Q}_{leg} - \hat{P}^i_{leg} = 0, \quad i = 1, \dots n.$$

For each t_i, these design equations are numerically minimized limb-by-limb, providing the kinematic skeleton and the inverse kinematics for each t_i.

Figure 4 summarizes this synthesis process for one of the arms. The kinematics equation ($\hat{Q}_{arm}$) defines the motion of an arm in terms of an spherical joint for the shoulder ($\hat{S}_{shld}$) and two revolute joints for elbow and wrist ($\hat{R}_{elb}, \hat{R}_{wrst}$)

$$\hat{Q}_{arm} = \hat{S}_{shld}(\theta_{a1}, \theta_{a2}, \theta_{a3})\hat{R}_{elb}(\theta_{a4})\hat{R}_{wrst}(\theta_{a5}).$$

Given a motion capture data sequence obtained from an existing human subject and based on the markers location, a set of coordinate frames $\hat{P}^i_{arm} : \{\hat{P}^i_{up}, \hat{P}^i_{low}, \hat{P}^i_{hnd}\}$ aligned with the limbs is extracted for each t_i, where $i = 0, \dots n$. More specifically, $\hat{P}^i_{up}$ is aligned with the upper arm, $\hat{P}^i_{low}$ with the lower arm and $\hat{P}^i_{hnd}$ with the hand. Combining the kinematics equation and the dataset, the design equations can be formulated as

$$\hat{Q}_{arm} - \hat{P}^i_{arm} = 0 \quad i = 1, \dots n.$$

These equations can be numerically minimized using a hierarchical strategy. First, the system solves for the spherical shoulder joint ($\tilde{S}_{shld}(\theta^i_{a1}, \theta^i_{a2}, \theta^i_{a3})$), then for the revolute elbow joint ($\tilde{R}_{elb}(\theta^i_{a4})$) and ends with the wrist revolute joint ($\tilde{R}_{wrst}(\theta^i_{a5})$). The kinematic skeleton remains unchanged over time ($\tilde{S}_{shld}$, $\tilde{R}_{elb}$, $\tilde{R}_{wrst}$), while the inverse kinematics (θ^i_{a1}, θ^i_{a2}, θ^i_{a3}, θ^i_{a4}, θ^i_{a5}) reflects the subject pose at t_i, always referred to the kinematic skeleton. The kinematic skeleton is compactly represented in terms of a point c for a spherical joint and as a rotation axis for a revolute joint, using a point c and a vector $\mathbf{s}$.

Figure 5 shows two sequences of inputs and results obtained for a synthetically generated reference data set (A) and a real motion-capture sequence (B). In both cases, each of the rows captures a different point in time, corresponding to frame numbers 43, 45 and 47 for dataset A and 350, 375 and 550 for dataset B. The total length of each of these

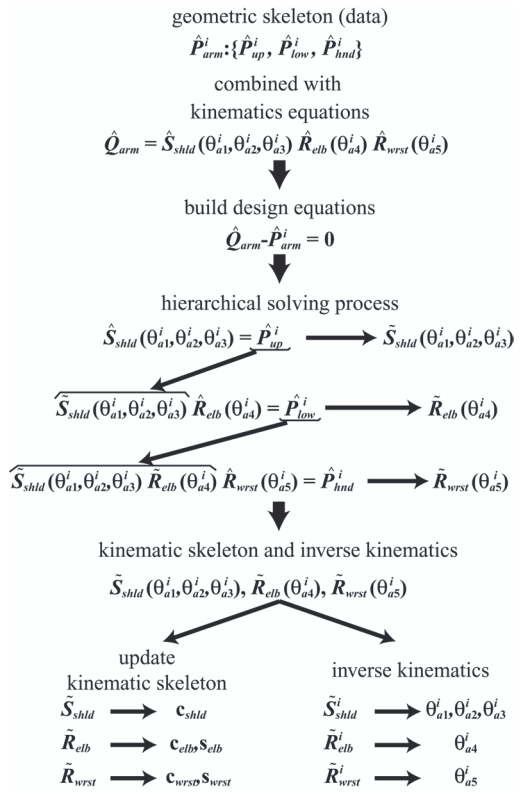


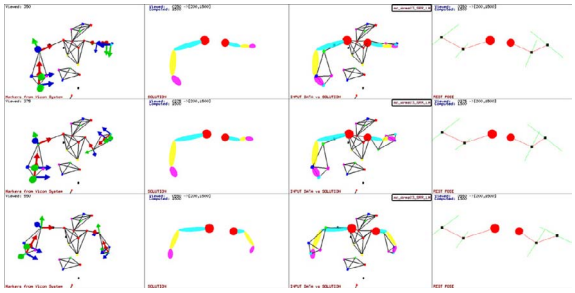
Figure 4. Hierarchical kinematic synthesis overview for an arm.

sequences was 80 for dataset A and 1800 for dataset B. The synthetic dataset was generated to present skeletal motion at all the joints. The physical dataset was acquired at a sampling rate of 120 Hz over 15 seconds resulting in 1800 frames. The data was passed to the synthesis algorithm and a stable solution was reached once all joints showed a sufficiently rich motion that enabled the synthesis for the complete model. For the discussed sequences, motion variations were already sufficient after frame 15 for dataset A and 3 for B. The first column shows the input motion capture data used for the motion synthesis process, the second the recovered motion, while the third one compares them blending both views. The last column contains the kinematic skeleton which should remain constant during the whole sequence.

In the case of the real motion capture dataset, we only recover the upper body and we can observe how noise in the captured data and the simplicity of our human model affect the stability of the results. The human motion synthesis module is successfully operating at interactive rates on synthetically generated datasets [10, 11, 9] and motion capture data. The use of dual quaternions for the specification of the synthesis problem facilitates a simpler and more

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

(a) Dataset A. Using a synthetic dataset as input.



(b) Dataset B. Using real motion capture data as input.

Figure 5. Examples of synthesized motion for different input datasets.

elegant formulation compared to the approaches using homogeneous transformations or exponential maps and twists. Major advantages of this methodology are that (i) no human intervention is required during the synthesis process, (ii) it is extensible to support more sophisticated joint types, (iii) it can synthesize the motion of articulated creatures with non-human structure, (iv) it provides for arbitrary subjects with the appropriate location of joints, length and twist angle of links forming the limbs, and (v) enables the creation of new animations for the constructed avatar.

4. Virtual Reality Platform

A flexible VR platform, called LIQÜID, was developed to provide our system with the required visualization and simulation capabilities. The platform was designed to run on PCs, supporting the use of consumer-level peripherals, as well as head mounted displays, gloves, 3D mice and multi-degrees of freedom sensors.

4.1. Platform Design

LIQÜID offers several features such as optimized rendering capabilities, stereoscopic viewing in various formats, dynamic link (DLL) support for custom modules, collision detection and response, and tracker data acquisition.

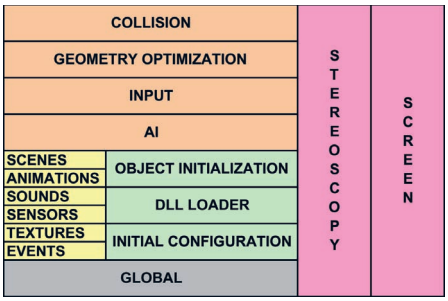


Figure 6. LIQÜID diagram.

Figure 6 shows LIQÜID's multilayered modular structure. The global system layer manages script files, mathematical operations and a list of objects describing the different items contained in a virtual environment. Supported object types are scenes, animations, sounds, sensors, textures and events. Of particular importance is that all events can be triggered using XML-based scripting files. Several loaders are included in the platform, allowing (a) events to be triggered based on the premises specified in a set of scripts and configuration files, (b) the 3D scenario and animations to be handled using static and dynamic geometry loaders, and integration of (c) texture management and (d) input devices loaders necessary for callback initialization supporting keyboards, mice and more specialized VR equipment.

At a different level, the platform also provides specific modules designed to improve the performance of the presented application. The modules comprise an AI layer intended to provide avatars with enough autonomy to reach their respective target location, using a path planning algorithm, geometry optimization processing to ensure real-time interaction, sensor data acquisition parsers, sound loaders, and collision handling. The last stage of the pipeline is the rendering layer, supporting monoscopic and stereoscopic viewing.

4.2. Real-time Rendering Platform

LIQÜID is optimized to ensure real-time interaction and rendering, using: (a) portal rendering, to determine the visible geometry from a given center of projection, (b) BSP (Binary Space Partitioning) trees speeding the selection of triangles to be rendered and their ordering with respect to the camera, (c) bounding box-based collision detection, (d) a reaction algorithm based on friction and gravity, and (e) lightmaps used to precompute static shadows. Our static geometry loader (Section 4.1) supports the ASE (ASCII Scene Exporter) file format, which allows environmental models or components to be created with commercial software tools. This also provides compatibility with a wealth of models and model collections that are readily available.

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

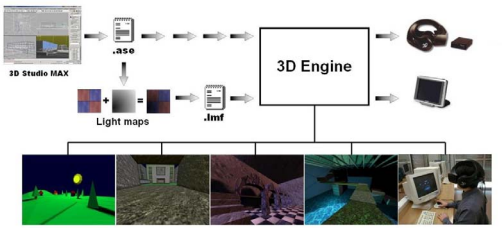


Figure 7. Data flow and representation.

Based on the static geometry describing a virtual world and spotlights present in the scene, static shadows are precomputed and stored as lightmaps. The render engine loads them independently or directly embedded in the scene file. For the second case, scenes are converted from ASE into an internal binary format. Figure 7 shows the data flow and data representation.

4.3. Basic data flow

The current LIQÜID prototype supports DLL loading, DirectX/OpenGL rendering, ASE and XML parsing, generic mathematical operations in assembler and C/C++ using internal data structures for data and memory management. Data exchange and flow between the core modules is illustrated in Figure 8. Arrows show the direction of communication among layers, while dotted lines indicate the existing data flows.

Root is the first module to be initialized in the main application procedure. Next, the XML parser configures the graphics engine and starts loading the scene geometry, sounds, movies, etc. This parser sends data to the Scene and Memory Managers, which store all information about the active objects. Once the Scene Manager has completed its task, the Render layer uses OpenGL or DirectX to perform the final rendering. To facilitate the scalability and extensibility of the design, DLL's are used to bind additional system features at runtime. The DLL Manager searches for the necessary libraries, loads them into memory and takes control of their inner operations. In addition, the Script Manager filters external files containing scripts which allow the specification of very simple actions to control and activate objects in the scene.

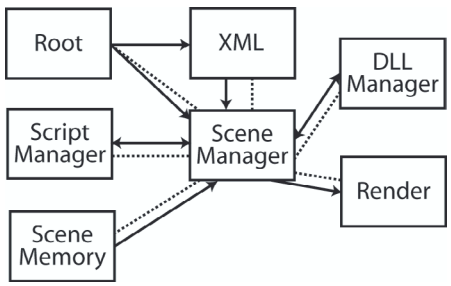


Figure 8. Basic data flow diagram of LIQÜID.

5. Risk Analysis and Prevention

Risk analysis and prevention techniques can be defined based of the current sensor data and the corresponding state of the analyzed environment. To support situation awareness a risk analysis and prevention architecture was developed (Figure 9), consisting of an Emergency Simulation Engine (ESE), Event Handler (EH), Alarm Generator (AG), Avatar Intention Generator (AIG) and Motion Path Planner (MPP).

Initially, scene geometry, animations, avatars and sensor data are dynamically loaded into the virtual environment, following the specifications available in a set of XML configuration files. These configuration files specify the virtual environment by listing the static geometry files (ASE files) and number of avatars in combination with their properties (initial location, leadership role and animation) and sensors. Each sensor in the scene has several attributes such as (a) a threshold specifying when its corresponding alarm should be triggered, (b) location in the scene, and (c) type such as humidity, temperature or acceleration. Once all of the data describing the elements in the virtual scenario are loaded, an Event Handler (EH) is started and timestamps each action simulating the dynamic behavior of sensors monitoring the building. Centralized control of the global operation of each simulation is then passed to the Emergency Simulation Engine (ESE). The ESE then periodically retrieves the current action from the EH and initiates the required simulation steps. First, it asks the Alarm Generator (AG) to update each alarm status according to the current sensor values and its specific threshold. The ESE also controls the Avatar Intention Generator (AIG). For each avatar in the scene, the AIG checks if the number of alarms has changed since the last event and decides the new destination cell for the avatar. Every time an avatar changes its destination cell, it also requests the Motion Path Planner (MPP) to optimize for the best path.

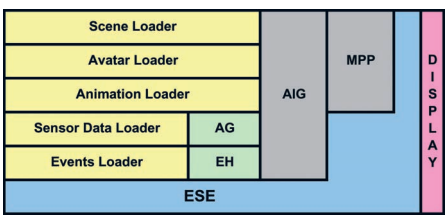


Figure 9. Risk Analysis and Prevention architecture.

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

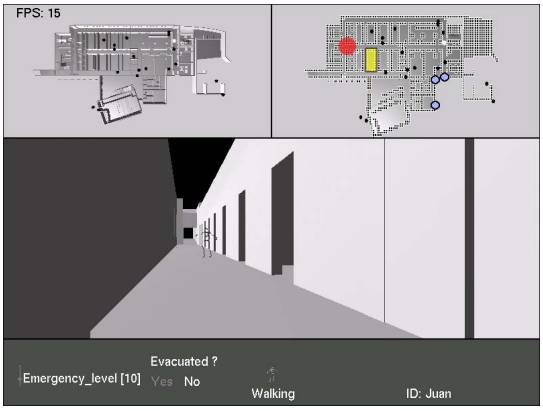


Figure 10. Interface design.

The graphical user interface is divided in four viewports (Fig. 10) including two general views of the scene, showing avatar and sensor locations (top views), a first person view for the currently selected avatar (middle view), and an information panel (bottom) describing the avatar emergency level, personal identifier, a description of the animation currently played (running, jumping, walking or rest pose) and an indicator showing the avatar's evacuation status. In addition, the top-left viewport shows a general 3D view of the scenario and represents each avatar as a dot, using white for the selected one (shown in the middle and bottom views) and black for the rest. In addition, the top-right viewport presents an orthographic view of the scenario together with its structural information, i.e., walls, emergency exits (light-blue circles), hazardous areas (yellow) and sensor/alarm locations (red) whenever they are active.

5.1. Emergency Simulation Engine (ESE)

The Emergency Simulation Engine performs several update operations on a fixed time step basis, and uses callbacks to continuously retrieve pending actions from the event handler, gather all available sensor information and generate corresponding alarms, updates avatar positions, destination cells and animations based on the AIG module and plan a new trajectory when needed, using the MPP module. It also generates some interface parameters necessary for a correct scene and avatar visualization.

5.2. Event Handler (EH)

As The Event Handler module processes and keeps track of the list of actions to simulate at each instant in time. These actions are distributed over time and specified either in a configuration file to evaluate synthetically generated events, or provided via a physical sensor network. The EH provides the ESE with a set of actions to execute for each time step.

5.3. Alarm Generator (AG)

The Alarm Generator processes all data sensor values provided by the acquisition system or a synthetically generated script, and decides if an alarm should be triggered for a specific sensor. Decisions are made based on a predetermined threshold associated with each sensor type.

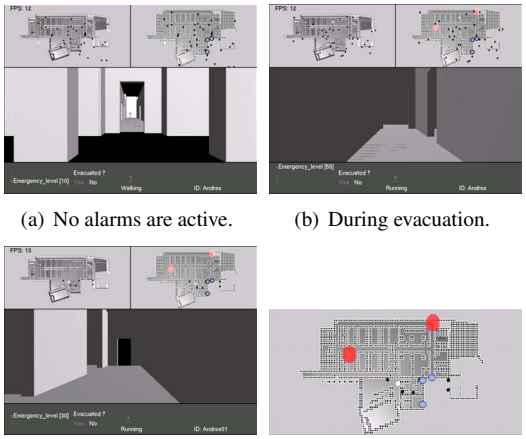
5.4. Avatar Intention Generator (AIG)

The Avatar Intention Generator determines the emergency level associated with each avatar. For a given avatar, this level provides a measure describing how risky the situation is depending on the environment, the distance to a destination cell and the number of activated sensor events or alarms. It also determines the currently displayed animation and the avatar's next target cell. Avatars respond to two different scenarios that can be classified as either a high-emergency level or a low-emergency level. The first case is activated when one or more alarms are triggered and the avatar is directed to the closest emergency exit. In the second situation, in absence of alarms, the avatar follows a predefined task or wanders randomly visiting only valid cells in the scene.

5.5. Motion Path Planner (MPP)

Once the current state of all the alarms is determined, each avatar in the scene has its own emergency level, animation pattern and destination cell. The Motion Path Planner provides each avatar with an obstacle-free path to follow, using a modified version of an A* algorithm with pruning [5]. Additionally to the classical algorithm, and to speed up the path-finding process, the path-tree is also pruned when the algorithm fails a predetermined number of times. Essential to the algorithm is the specification of the scenario's structural elements. Each floor is represented using a highly reconfigurable cell-based matrix data structure, where each cell contains a tag describing its contents and/or function, such as, sensor type, wall, furniture and emergency exit. A new path is planned only when needed. That is, once a situation triggers a change in the global state of the alarms, either from none to one or more alarms activated, or deactivated. Updates also occur when a wandering avatar reaches its target destination. The MPP takes the avatar location in the scene as the starting cell, and finds an optimum cell-based trajectory that connects it either to the closest emergency exit or to a random cell, as stated by the AIG.

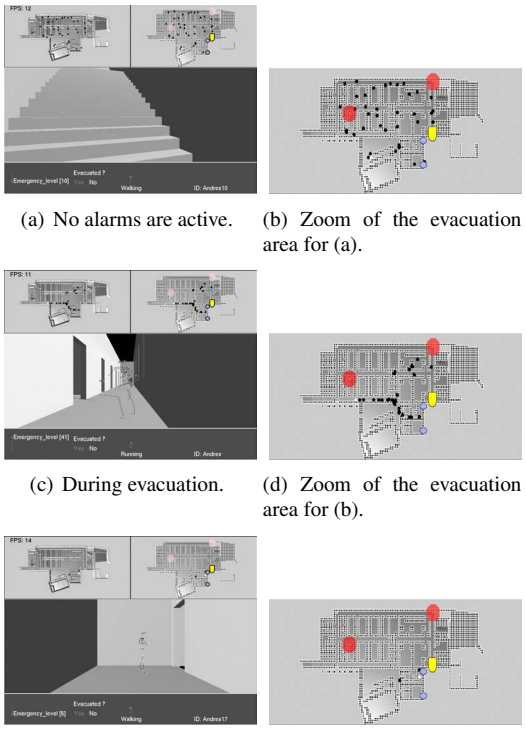
Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY



(a) No alarms are active. (b) During evacuation.

(c) Building almost evacuated. (d) Zoom of the evacuation area for (c).

Figure 11. Case Study A: 40 avatars, 2 alarms, 20 sensors, 4 emergency exits, 15 frames per second (average).



(a) No alarms are active. (b) Zoom of the evacuation area for (a).

(c) During evacuation. (d) Zoom of the evacuation area for (b).

(e) Building almost evacuated. (f) Zoom of the evacuation area for (c).

Figure 12. Case Study B: 40 avatars, 2 alarms, 20 sensors, 4 emergency exits, 15 frames per second (average).

5.6. Case Studies

The first floor of the Calit2 building at UC Irvine (Fig. 2) was used for a set of case studies, evaluating the performance of the risk analysis and prevention system. To study avatar behavior, these scenarios were designed using different spatial arrangements and event triggers. The objective was to identify possible improvements to the environment and evacuation plans that would further improve the evacuation of building inhabitants. Case studies included:

A crowded area and a sudden alarm triggered in the same building area A local emergency arises in a specific location where most of the building inhabitants are concentrated. This case study analyzes a scenario that assumes that a fire or toxic leak was detected at a specific location, and has not yet propagated to the rest of the building. For the case of a low intensity earthquake affecting the building, an emergency may be generated only in hazardous areas such as laboratories where chemicals are stored and manipulated. The motion path planner (MPP) operates differently depending on the factors taken into account when determining the path each avatar follows in case of an emergency. If the density of avatars on each cell and the location of hazardous areas (e.g. laboratories storing chemicals) are not considered, all avatars evacuate the building using the same emergency exit. In case these factors are considered, some avatars escape using the closest emergency exit and the rest use the most suitable emergency exist avoiding the areas labeled as hazardous. Evacuation speed of the avatars depends on the emergency level. Figure 11 shows this scenario with a total of 40 animated avatars, four emergency exits and 20 sensors that triggered an alarm next to a hazardous area ($time$ =1'10"). The building was completely evacuated at minute 3'35" and the complete simulation was visualized at an average frame rate of 15 frames per second.

Crowded area and an alarm in a different area of the building Given the distance between the crowded avatars area and the location of the alarm, part of the population may not be fully aware of the danger and does not take any action to evacuate the building, even though the building is expected to have a global evacuation plan that is executed. Avatars are directed to their closest emergency exit, naturally avoiding dangerous areas. The avatars danger perception increases over time as the toxic leak or fire spreads.

A dispersed crowd and an alarm set in one specific area Those avatars close to the alarm rapidly use the nearest emergency exit and the rest, depending on how close they are, flee based on meeting other avatars who are evacuating the building. Assuming the crowd is dispersed, the emergency perception of each avatar may vary greatly based on

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE
Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY

the avatars they meet during evacuation.

Dispersed crowd and multiple alarms in different locations This scenario recreates the most chaotic situation in which a serious disaster affects the building. Avatars are conscious of the great danger and sometimes make mistakes while choosing the best route towards the emergency exit. Bottlenecks and continuous path corrections complicate the evacuation and highlight features of the building floor plan distribution that can inevitably prevent some avatars from abandoning the building. Figure 12 shows this scenario with a total of 40 animated avatars, four emergency exits and 20 sensors that concurrently triggered two alarms ($time$ =1'10"). The building was completely evacuated at minute 3'16" and the complete simulation was visualized at an average frame rate of approximately 15 frames per second.

6. Conclusions

With the California Institute for Telecommunications and Information Technology (Calit2) building at UCI, a massively instrumented environment was selected, providing access to sensor networks tasked with monitoring the performance of the building under normal as well as unusual (earthquakes, etc) conditions. An open visualization platform was developed to support the study of risk assessment and mitigation plans. This platform allows different algorithms to be integrated, evaluated, and compared supporting the analysis of different risk mitigation techniques. The platform is designed to be scalable and highly reconfigurable via script files, allowing for the direct comparison of different scenarios. The motion path planning behavior can be reconfigured to handle different sets of considerations such as avatars adopting a leadership role, distance to alarm sources and hazardous areas.

Analyzing the simulation results helps to minimize risks through redesign of the environment. Some examples include, deficiently monitored zones that can be modified using a different sensor placement, additional emergency exit paths can be identified to facilitate evacuation of crowded areas, and redistribution of hazardous areas such that they do not interfere with the main evacuation corridors.

Avatars are used within this virtual environment to better articulate the response of humans within the space. Biomechanical constraints can be modeled on a per avatar basis, and are crucial for the avatar intention generator (AIG) to faithfully recreate multiple avatar reactions to different emergency situations. Avatars do not necessarily react in a logical manner to a disaster and in very severe scenarios tend to panick and group themselves, becoming indecisive. Using these simulations, first responder crews can evaluate and predict possible bottleneck locations where, for example, furniture can obstruct evacuation. Non-planned situations can arise from this analysis, facilitating the evaluation and design of fallback plans.

7. Acknowledgments

This research was supported in part by the California-Catalonia Program for Engineering Innovation, the Balsells Fellowship Program and the California Institute for Telecommunications and Information Technology (Calit2) program. The above support is greatly appreciated.

References

[1] N. I. Badler, C. A. Erignac, and Y. Lu. Virtual humans for validating maintenance procedures. *Communications of the ACM*, 45(7):56–63, July 2002.

[2] N. I. Badler, C. B. Phillips, and B. L. Webber. *Simulating humans: computer graphics, animation, and control*. Oxford University Press, 1992.

[3] R. Bukowski and C. Séquin. Interactive simulation of fire in virtual building environments. In *SIGGRAPH '97*, pages 35–44, 1997.

[4] C. Esteves, G. Arechavaleta, and J.-P. Laumond. Motion planning for virtual mannequins cooperation. Technical Report 04574, LAAS-CNRS, 2004.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, SSC4(2):100–107, 1968.

[6] T. A. Nüssle, A. Kleiner, and M. Brenner. Approaching urban disaster reality: The ResQ firesimulator. Technical Report 200, Institut für Informatik Universität Freiburg, April 2004.

[7] N. Schurr, J. Marecki, M. Tambe, P. Scerri, N. Kasinadhuni, and J. P. Lewis. The future of disaster response: Humans working with multiagent teams using DEFACTO. In *AAAI Spring Symposium on Homeland Security*, Stanford, CA, March 2005.

[8] I. Takeuchi, S. Kakumoto, and Y. Goto. Towards an integrated earthquake disaster simulation system. In *First International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, Padova, Italy, July 2003.

[9] M. Villa-Uriol. *Video-Based Avatar Reconstruction and Motion Capture*. PhD thesis, Henry Samueli School of Engineering, University of California, Irvine, 325 Engineering Tower, Irvine, CA, 2005.

[10] M. Villa-Uriol, F. Kuester, and N. Bagherzadeh. Hierarchical kinematic synthesis of motion for avatars creation. In *International Conference on Computer Animation and Social Agents (CASA)*, 2005.

[11] M. Villa-Uriol, F. Kuester, N. Bagherzadeh, A. Perez, and J. M. McCarthy. Kinematic synthesis of avatar skeletons from visual data. In J. Lenarčič and C. Galletti, editors, *9th International Symposium on advances in robot kinematics*, pages 171–180. Kluwer Academics Publishers, June 2004.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on October 26,2024 at 10:18:18 UTC from IEEE Xplore. Restrictions apply.
IEEE COMPUTER SOCIETY