Chapter 20

USING 3D MODELS AND DISCRETE SIMULATIONS IN INFRASTRUCTURE SECURITY APPLICATIONS

Pierluigi Assogna, Glauco Bertocchi, Alberto Paoluzzi,
Michele Vicentino, Giorgio Scorzelli and Roberto Zollo

Abstract Next generation systems for critical infrastructure protection must support capabilities such as behavior analysis, situation modeling and data mining integrated within sophisticated virtual or augmented reality interfaces. This paper describes the design goals and implementation of a platform for critical infrastructure security applications. The platform is designed to support semi-automated 3D modeling of infrastructures, 3D integration of sensor networks, situation modeling and visual simulation via 3D animation, and advanced situation analysis. Such a system would enable operators to recognize preliminary indications of crisis situations and promptly activate the appropriate countermeasures. It would also assist them in optimizing normal operations and conducting simulations for emergency planning and crisis management.

Keywords: Geometric modeling, simulation, infrastructure security

1. Introduction

Visual presentations can offer excellent situation awareness for security applications. An advanced platform, providing a 3D geometric model of the system to be protected and facilities for event discovery, tracking and situation evaluation, is an ideal candidate. It could serve as point of reference for integrating vision, sensor, tracking and security systems designed for infrastructure protection. It would provide a reliable basis for high-level situation awareness and support coordinated and optimized decision making.

Critical transport infrastructures that straddle national borders, such as tunnels, bridges, railways hubs and airports, require a novel security approach. Their ability to address threats and react to crises relies on the strong coordination of operational activities. This is because security threats may arise

Please use the following format when citing this chapter:

Assogna, P., Bertocchi, G., Paoluzzi, A., Vicentino, M., Scorzelli, G. and Zollo, R., 2008, in IFIP International Federation for Information Processing, Volume 290; *Critical Infrastructure Protection II*, eds. Papa, M., Shenoi, S., (Boston: Springer), pp. 269–278.

not only from malicious attacks but also from natural events (storms, floods, earthquakes) and unexpected events (traffic congestion, accidents). Security should always be maintained and the systems should be able to deal with all the facets of the problem. For this reason, an advanced security system must be able to infer the consequences of events based on the information received and the inferences should be reliable even when some of the information is missing.

Virtual or augmented reality interfaces [14, 21] can be very useful. Consider, for example, a building on fire or a railroad during a storm, where video feeds are not available. An advanced security system could use virtual/augmented reality to reproduce the situation and support optimal decision making through modeling and simulation. However, existing security systems are typically clever assemblies of sensor subsystems with limited capabilities for assisting personnel during normal operations and crises. Moreover, they are not designed to be used when responsibility is shared between different organizations as in the case of transnational critical infrastructures.

This paper describes the development goals and implementation directions of a new platform for infrastructure security applications. The platform is intended to provide: (i) (mostly) automatic 3D modeling of infrastructures; (ii) 3D integration of sensor networks, including video surveillance; (iii) situation modeling though hierarchical event graphs annotated with actor models and behavior; (iv) visual simulation via 3D animation; (v) situation analysis based on the divergence of sensor feedback from evolving simulations; and (vi) weak signals of risky situations deduced by comparing the results of situation analysis with a knowledge base of past events and scenarios. This paper also discusses the use of geometric modeling and concurrent programming tools that support sophisticated modeling, simulation and analysis functionality.

2. Awareness Paradigm

Current security systems are useful for providing alerts and alarms based on signals from sensor networks. However, they do not offer the high-level functionality needed to accomplish complex tasks such as: (i) identifying unusual and potentially dangerous or difficult circumstances, (ii) recognizing the weak signals of risky situations, (iii) promptly activating countermeasures, and (iv) assisting personnel in the optimized management of crises. An awareness platform with this functionality can be created by integrating:

- 3D geometric models of an infrastructure that are used as the basis for all information collected about the infrastructure.
- An interoperable knowledge framework that captures information about the infrastructure along with its behavior, events, users, resources, etc., which can be used as a basis for statistical and situational models.
- Pre-existing heterogeneous sensor and security systems.
- Virtual or augmented reality interfaces and advanced gaming techniques.

A critical infrastructure surveillance system cannot limit itself to a representation of what is happening; it should also provide tools for analyzing the

patterns, rhythms and interplays of events, and for anticipating future impact. This enables automatic as well as human responses to be more effective and proactive. Software agents provide a useful abstraction for a security platform: the agents perform all the major activities (e.g., sensing the environment, acting on devices and alerting security personnel). In particular, we imagine a "holonic," multi-level organization of agents that maintain and use an awareness base represented by the integration of models and sensing, surveillance and control functions. Note that a "holon" [10] is a component of a system that can work autonomously even if detached from it, and that works well in cooperation with other holons. According to this vision, the security platform could represent the intelligence of the controlled infrastructure, almost like a living organism endowed with self-consciousness [10].

A security platform should enforce modeling capabilities to the maximal extent; this is because all the support provided would be grounded on model-based simulations. Permanent situation awareness would be achieved using techniques that model an infrastructure, simulate its normal behavior, acquire and track normal and abnormal events, understand the weak signals that indicate a crisis and react to them in an optimal manner. The ability to simulate events, actions and reactions would enable the security platform to maintain the critical infrastructure in a safe state. Operations personnel could interact with the platform, adding experience and judgment that cannot be coded or simulated.

To implement such a platform, we are employing a novel approach for symbolic situation modeling. The approach uses (machine- and human-readable) executable and combinable storyboards [2]. The implementation engages the Erlang concurrent programming language [1] integrated within an advanced 3D modeling environment based on the geometric language PLaSM [15] and augmented with gaming engines.

3. Spatial Modeling

Howarth [9] has shown that spatial models play a key role when interpreting a dynamic, uncertain world for surveillance applications. We surveyed a number of works dealing with spatial models and their use in security applications. Based on this research, we selected the cellular decomposition of space as a foundation for supporting visual surveillance applications. We use the geometric language PlaSM to handle geometric information. Figure 1 presents the virtual model of a warehouse district produced using a few pages of PLaSM code. The runtime evaluation of submodels of buildings or portions of buildings produces the 3D scenes.

The design language PLaSM is a geometry-oriented extension of FL, a functional language developed by Backus' group at IBM [4]. Building on the paradigm proposed by Backus in his Turing Award Lecture [3], FL introduces an algebra over programs that provides several interesting features. In particular, existing programs are combined very elegantly to create new programs. Programs that are equivalent to and simpler than the original programs can be

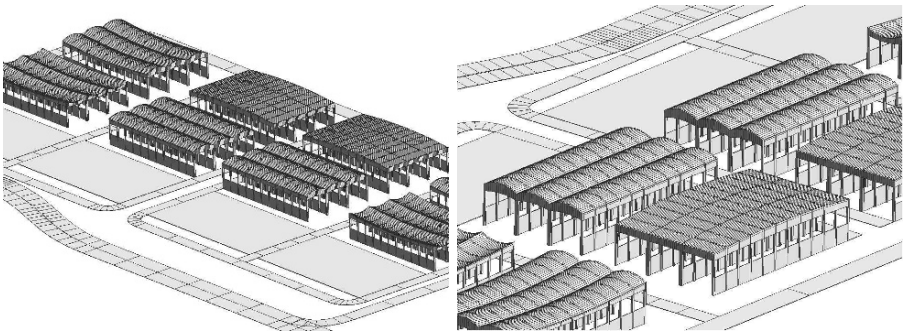

Figure 1. Virtual model of a commercial district.

created at design time or compilation time. Significant benefits in programming style and efficiency are achieved through the use of program prototyping.

Primitive objects in FL include characters, numbers, truth values and polyhedral complexes. A polyhedron is a quasi-disjoint union of polytopes (bounded convex sets). Expressions are either primitive objects, functions, applications or sequences. According to the FL semantics, arbitrary PLaSM scripts can be written using only three programming constructs:

- **Application:** The application `f:x` of function `f` to the value `x` of an input parameter (element of the function domain) produces an output value in the function codomain.
- **Composition:** The composition of functions `(f~g~h):x` ≡ `(f~g):(h:x)` ≡ `f:(g:(h:x))` produces the pipelined execution of their reversed sequence (Figure 2).
- **Construction:** The construction of a vector function `[f,g,h]` produces the parallel execution of its component functions `[f,g,h]:x` ≡ `<f:x, g:x,h:x>` (Figure 2).

4. Model Generation

We capitalize on a novel parallel framework [6] for high-performance solid and geometric modeling that compiles the generating expression of a model into a dataflow network of concurrent threads, and splits the model into fragments that are distributed to computational nodes and generated independently. Progressive BSP trees are used for adaptive and parallelizable streaming dataflow evaluation of geometric expressions, and associated with polyhedral cells of hierarchical polyhedral complex (HPC) data structures used by the PLaSM language.

The security platform design makes strong use of spatial models produced by the cellular decomposition of structures (buildings, tracks, lanes, tunnels, bridges, etc.) from 2D plans. Figure 3(a) shows a 2D map of the Palatino Hill

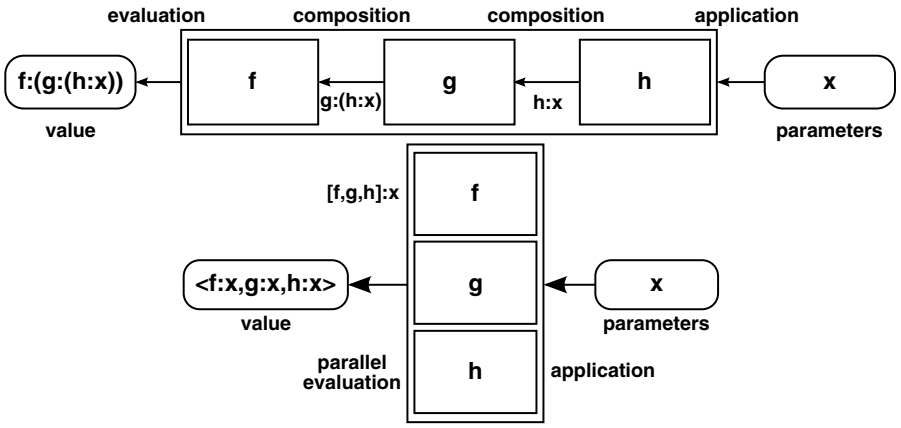

Figure 2. Main paradigms of the PLaSM language.

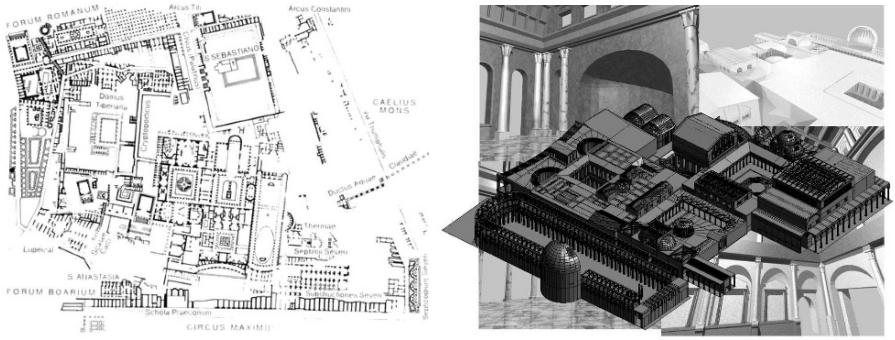

Figure 3. (a) Palatino Hill (2D map); (b) *Domus Augustana* (3D PLaSM image).

in Rome. Figure 3(b) presents the 3D PLaSM reconstruction of the emperor's palace (*Domus Augustana*), a site endowed with the strongest security system known in ancient times. Note the enhanced level of awareness provided by the 3D model compared with the 2D model.

We are currently working on the automatic generation of PLaSM models from line drawings in the Autocad format [2]. A geometric representation of an infrastructure or portion of an infrastructure, progressively generated at increasing levels of detail, will be produced at runtime by a streaming dataflow process.

The generation of models of complex buildings from plans is a difficult and largely unsolved inverse problem [11]. The difficulty of reconstructing 3D buildings from architectural plans is confirmed by the rarity of scientific literature on the topic (see, e.g., [9, 11]). However, a fast semi-automatic solution has been proposed by Paoluzzi and Scorzelli [17]. Their procedure is as follows. First,

Autocad files with line-drawings of 2D architectural plans are transformed into double interval trees and quad trees in order to efficiently answer interactive proximity queries. Next, semantics is assigned to small line subsets via pattern-based recognition of the components of the building fabric (internal partitions, external enclosures, vertical communication elements, etc); this is subsequently translated into PLaSM scripts. These symbolic scripts are then evaluated to produce streaming solid models at various levels of detail or adjacency graphs of the critical infrastructure or portions of the infrastructure.

5. Model Simulation

Modeling techniques tend to be found in isolated communities [18]: geometric models are used in computer-aided design (CAD) and computer graphics; dynamic models in computer simulations; and information models in information technology applications. When models are not included within a single digital environment, methods for connecting them seamlessly are generally not established. One of the major challenges we faced in our research was to identify the best abstraction for integrating these three different modeling areas.

According to [13], discrete concurrent systems may be modeled and simulated using three paradigms: event graphs and PERT (event scheduling), bipartite graphs and Petri Nets (activity scanning), and finite-state systems and Markov chains (process interaction). Network programming techniques [5] may be used to animate complex simulations. Here, the behavior of each actor is described by a path in an activity network that codifies the storyboard as a directed acyclic graph and describes causal and temporal relationships between events. Each storyboard arc may be associated with a spline in the configuration space of the actor, which models fine behavior permitted by its degrees of freedom. This model is hierarchical, since each arc may be substituted by its (local) storyboard and used to decompose a macro event to a finer level of detail. Also, the parameters of the probability distributions of stochastic variables of interest (e.g., the most likely time or the lead/lag time of an event) may be computed. The depth of the event hierarchy may vary locally depending on the complexity of events and the degree of realism of a simulation [16].

Petri nets [20] may be used to mathematically describe the evolution of a concurrent system. This approach uses networks with two types of nodes (places and transitions) and arcs where several types of tokens may move within the network. The state of the modeled system is given by the distribution of tokens in places. The annotations of fine behavior are described by splines in configuration space, defined by discrete sets of points consisting of at least two known configurations (in the case of linear behavior). Petri nets may be hierarchical and timed. The hierarchical communicating and concurrent state machine (HCSM) framework may be used to plan the behavior of reactive synthetic agents in interactive simulation environments [7]. The HCSM framework is well suited to modeling the reactions of autonomous agents and directing them to produce the desired simulations.

6. Architectural Issues

The main architectural goal is to integrate advanced modeling, simulation, virtual or augmented reality, and gaming and tracking techniques in a centralized framework. The intent is to provide a platform where the tasks involved in securing a critical infrastructure can be evaluated and optimized, and where security personnel are provided with high-level decision-making tools, including interoperable tools for situation modeling and simulation.

6.1 Concurrent Programming

The implementation employs PLaSM for geometric programming and Erlang for simulation. Erlang [1] is a concurrent functional programming language and runtime system, which is characterized by strict evaluation, single assignment and dynamic typing. Strict, or eager evaluation, means that expressions are evaluated as soon as they are bound to variables and the values are subsequently recalled without evaluation. The language is purely functional, without the mutable state that is induced by multiple assignments. Thus, the code is easy to understand and facilitates correctness proofs. Dynamic typing means that type checking is performed at runtime.

Erlang adopts a mathematical model of computation that uses "actors" as universal primitives for concurrent computing [8]. The actor model has been used both as a theoretical framework for concurrency as well as the basis for several practical concurrent systems. An actor can make local decisions, create other actors, send and receive messages, and determine how to respond to messages. Data structures, functions, semaphores, monitors, ports, descriptions, logical formulas, numbers, identifiers, demons, processes, contexts and databases are all special cases of actors, whose behavior is described by message sending.

6.2 Event Processing

The platform is designed to use complex event processing (CEP) techniques implemented using Erlang. CEP integration will employ a state-of-the-art architecture for stream event processing in complex computer systems [12, 19]. Different events, combined with different behavior and environments, will elicit adequate responses from the platform so that, even when unattended, it can perform actions to avoid or minimize unwanted scenarios. Optimal control will be achieved, of course, by combining automatic responses with the experience and judgment of experts.

This approach to security awareness, providing sensor fusion as well as event discovery, registration and interpretation, will make strong use of spatial models. As discussed earlier, these models are produced as cellular decompositions of structures generated from PLaSM representations created semi-automatically from 2D line drawings.

The platform will employ advanced techniques for interactive visualization of, and user interaction with, real-time and recorded surveillance data. Research results in the areas of visual perception, 3D interactive computer graphics, virtual reality and gaming will be incorporated. State-of-the-art techniques for 3D navigation and spatial montage of streaming sensor data will be implemented to produce a task-specific interface optimized for situational awareness.

6.3 Management Support

The platform will provide support for workforce management, especially in the case of crises that require significant manual intervention. The tasks to be supported will range from routine camera surveillance by operators to *ad hoc* crisis management by decision makers. Different types of users will be able to access and interact with relevant information provided by the underlying databases and (generated) knowledge.

Flexible, task-specific user interfaces will support the collection, retrieval and presentation of information. A set of guidelines and components for implementing highly effective, task-specific user interfaces will be developed. These interfaces will enable automated (software-agent-based) actions as well as human actions directed from an operations control center. In particular, the interfaces will provide semantically-rich support based on the "newspaper," "agenda," "map," "telephone" and "television" metaphors.

7. Conclusions

Securing critical infrastructures requires sophisticated systems that provide capabilities such as behavior analysis, situation modeling and data mining integrated within virtual or augmented reality interfaces. Our advanced situational awareness system is designed to support semi-automated 3D modeling of infrastructures, 3D integration of sensor networks, situation modeling and visual simulation via 3D animation, and advanced situation analysis. In addition to offering sophisticated functionality, the system is designed to be highly scalable to handle large, complex infrastructures. Our paradigmatic reference for awareness, simulation and control of normal and abnormal situations is applicable to operation planning, security enforcement, crisis management and training. It can also be used within product lifecycle management platforms in aerospace, automotive and manufacturing industries, where geometric information is central to collaborative activities ranging from production to marketing.

References

[1] J. Armstrong, *Programming Erlang: Software for a Concurrent World*, Pragmatic Bookshelf, Raleigh, North Carolina, 2007.

[2] P. Assogna, G. Bertocchi, A. Paoluzzi, G. Scorzelli and R. Zollo, From 2D plans to 3D building models for security modeling of critical infrastructures, to appear in *International Journal of Shape Modeling*, 2008.

[3] J. Backus, Can programming be liberated from the von Neumann style? A functional style and its algebra of programs, *Communications of the ACM*, vol. 21(8), pp. 613–641, 1978.

[4] J. Backus, J. Williams and E. Wimmers, An introduction to the programming language FL, in *Research Topics in Functional Programming*, D. Turner (Ed.), Addison-Wesley Longman, Boston, Massachusetts, pp. 219–247, 1990.

[5] C. Bajaj, C. Baldazzi, S. Cutchin, A. Paoluzzi, V. Pascucci and M. Vicentino, A programming approach for complex animations (Part I: Methodology), *Computer-Aided Design*, vol. 31(11), pp. 695–710, 1999.

[6] C. Bajaj, A. Paoluzzi and G. Scorzelli, Progressive conversion from B-rep to BSP for streaming geometric modeling, *Computer-Aided Design and Applications*, vol. 3(5), pp. 577–586, 2006.

[7] J. Cremer, J. Kearney and Y. Papelis, HCSM: A framework for behavior and scenario control in virtual environments, *ACM Transactions on Modeling and Computer Simulation*, vol. 5(3), pp. 242–267, 1995.

[8] C. Hewitt, P. Bishop, I. Greif, B. Smith, T. Matson and R. Steiger, Actor induction and meta-evaluation, *Proceedings of the First Annual ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, pp. 153–168, 1973.

[9] R. Howarth, Spatial models for wide-area visual surveillance: Computational approaches and spatial building-blocks, *Artificial Intelligence Review*, vol. 23(2), pp. 97–155, 2005.

[10] A. Koestler, *The Ghost in the Machine*, Arkana, London, United Kingdom, 1967.

[11] R. Lewis and C. Sequin, Generation of 3D building models from 2D architectural plans, *Computer-Aided Design*, vol. 30(10), pp. 765–779, 1998.

[12] D. Luckham and B. Frasca, Complex Event Processing in Distributed Systems, Technical Report CSL-TR-98-754, Computer Systems Laboratory, Stanford University, Palo Alto, California, 1998.

[13] J. Miller, G. Baramidze, A. Sheth and P. Fishwick, Investigating ontologies for simulation modeling, *Proceedings of the Thirty-Seventh Annual Symposium on Simulation*, pp. 55–63, 2004.

[14] R. Ott, M. Gutierrez, D. Thalmann and F. Vexo, Advanced virtual reality technologies for surveillance and security applications, *Proceedings of the ACM International Conference on Virtual Reality Continuum and its Applications*, pp. 163–170, 2006.

[15] A. Paoluzzi, *Geometric Programming for Computer-Aided Design*, John Wiley and Sons, Chichester, United Kingdom, 2003.

[16] A. Paoluzzi and A. D'Ambrogio, A programming approach for complex animations (Part II: Reconstruction of a real disaster), *Computer-Aided Design*, vol. 31(11), pp. 711–732, 1999.

[17] A. Paoluzzi and G. Scorzelli, Pattern-driven mapping from architectural plans to solid models of buildings, presented at the *Israel-Italy Bi-National Conference on Shape Modeling and Reasoning for Industrial and Biomedical Applications*, 2007.

[18] M. Park and P. Fishwick, Integrating dynamic and geometry model components through ontology-based inference, *Simulation*, vol. 81(12), pp. 795–813, 2005.

[19] L. Perrochon, W. Mann, S. Kasriel and D. Luckham, Event mining with event processing networks, *Proceedings of the Third Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 474–478, 1999.

[20] J. Peterson, Petri nets, *ACM Computing Surveys*, vol. 9(3), pp. 223–252, 1977.

[21] I. Sebe, J. Hu, S. You and U. Neumann, 3D video surveillance with augmented virtual environments, *Proceedings of the First ACM SIGMM International Workshop on Video Surveillance*, pp. 107–112, 2003.