RAIRO Operations Research
RAIRO Oper. Res. **38** (2004) 89–103
DOI: 10.1051/ro:2004014

MODELING AND SIMULATION WITH AUGMENTED REALITY

Khaled Hussain[1] and Varol Kaptan[2]

Abstract. In applications such as airport operations, military simulations, and medical simulations, conducting simulations in accurate and realistic settings that are represented by real video imaging sequences becomes essential. This paper surveys recent work that enables visually realistic model constructions and the simulation of synthetic objects which are inserted in video sequences, and illustrates how synthetic objects can conduct intelligent behavior within a visual augmented reality.

1. Introduction

Simulation is widely used to model real systems under varying synthetic conditions. It can be used to predict the capabilities of a system which is being designed, or to predict the performance of a system which is being modified. Simulation often concentrates on the algorithmic description and control of synthetic entities which are being modeled, and modern discrete event simulators often use a graphical interface to enrich the user's interaction with the simulation runs. Major progress in simulation technology arises if one can evaluate synthetic conditions in realistic settings, and ask questions about "what would happen if ..." in a real environment and in the presence of some actual events.

This paper surveys some recent advances in this area, including the simulation of synthetic objects which are inserted in real video sequences, and the techniques

[1] School of Computer Science, University of Central Florida Orlando, Florida 32816, USA; e-mail: `khaled@cs.ucf.edu`

[2] Department of Electrical & Electronic Engineering, Imperial College, London SW7 2BT, UK; e-mail: `v.kaptan@imperial.ac.uk`

that can be used so that synthetic objects conduct seamingly intelligent behavior within a visual setting. The visually oriented work we summarize was reported at specialized conferences [11, 24], and the research concerning synthetic behaviours was presented in previous papers [16, 17, 26, 27]. A detailed description of our research in this area is reported in [25].

2. Visual augmented reality

Visual augmented reality creates a combination of a real and virtual scenes in which the user perceives a significant difference between the real and augmented world. The real scene (RS), as seen from a given viewing location and direction (the "aim-point"), is viewed through an appropriate video camera or sensor. We refer to the real world as a "scene", even though we are dealing with a video sequence representing the world as it is being viewed in real-time. The scene has different visual representations depending on where it is being viewed from, and the viewing point is often referred to as the "aim-point".

One of the difficult technical issues in augmented reality is the "registration problem", which refers to the need for determining the isomorphism between objects and features in a live scene with the corresponding features and the corresponding objects in an augmented version of that scene. Errors in registration will generate visual inconsistencies between real and virtual images with obvious consequences on the value of the augmented reality system for simulation purposes. Therefore several authors have addressed this issue, and in [3] it is indicated that registration using only information from a sensor based tracking system cannot achieve a perfect match. Thus several authors make use of an image processing based algorithm for improving registration. One approach is to detect features in the real image and uses them to enforce registration, while others have considered placing special marks (*e.g.* LEDs [3], circles [30], a calibration grid [28]) in the environment. Image-processing algorithms detect the locations of these marks and use them to enforce registration, assuming that one or more special marks are visible at all times. Another approach [30] uses a survey of the live environment with real-time instrumentation, providing more information about objects and their distances in the live environment, but requires specific equipment and significant amounts of additional computation for the interpretation of the sensors' output.

Almost all augmented reality techniques assume that virtual objects and live objects have the same detailed shape. This assumption is only valid for rigid objects such as roads and buildings: many virtual object generators will use a simplified representation, and will even sometimes only make use of templates; *e.g.*, a synthetic pine tree may be some idealized template of a pine tree, rather than the actual pine tree being represented at some location in a scene.

Figure 1 shows a simple example of a virtual vehicle behind a virtual tree. If we use current techniques (register the real tree with the virtual tree; then copy the visible part of the virtual vehicle into the live image), visual inconsistencies will happen as shown in Figure 1b. Also current techniques cannot tolerate any small

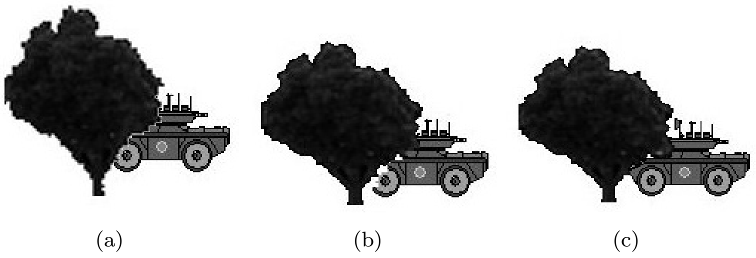


FIGURE 1. A simple example of a virtual vehicle behind a tree for illustrating the limitations of current techniques. (a) A virtual vehicle behind a virtual tree. (b) Inserting the virtual vehicle into the real scene using current techniques causes visual inconsistencies. (c) Inserting the virtual vehicle into the real scene using our technique does not cause visual inconsistencies.

errors in registration. Figure 2b and 2d shows an example of visual inconsistencies due to small errors in registration. To address this problem, we segment the live image into objects and then register them with the corresponding virtual objects.

We use neural network models derived from the "random neural network" (RNN) introduced in [5, 12–15], and developed in [18, 19, 22, 23] to aid in the registration. Other imaging applications of the RNN can be found in [21] where the texture based segmentation technique is introduced, and in [8, 9, 20] where its is applied to image and video compression. In [10] the RNN, together with adaptive contours, is applied to tracking moving objects in video sequences. This segmentation based approach is postulated on the premise that the real world is far more dense in significant objects, than in the number and type of synthetic objects we insert into it. This is of course totally different than an interactive computer game in which no real objects exist (*i.e.* everything is artificial), and in which there may be a large number of moving synthetic objects (*e.g.* aircraft, robots, etc.).

Our system design is based on the assumptions that the 3-D real world of stationary objects, *i.e.*, non-moving such as landmarks, terrain elevation, houses, trees, etc. is represented in a terrain database (TB) which has some desired level of precision. We also assume that there are a limited number of moving real objects and virtual objects, which together cover only a small fraction of the scene being viewed at any time or being represented in the TB, and that moving virtual objects are also represented in the TB. They can be synthetically moved, either manually (by the operator or user), or using a discrete event simulator. Using this approach, from the TB and with a specific viewing location and direction, it is possible to synthetically generate a graphics image representing the scene as it is viewed from the TB. This graphics based scene (GS) will include the synthetic objects. Inaccuracies in the TB are expected to exist and will often have to be compensated for in real-time.

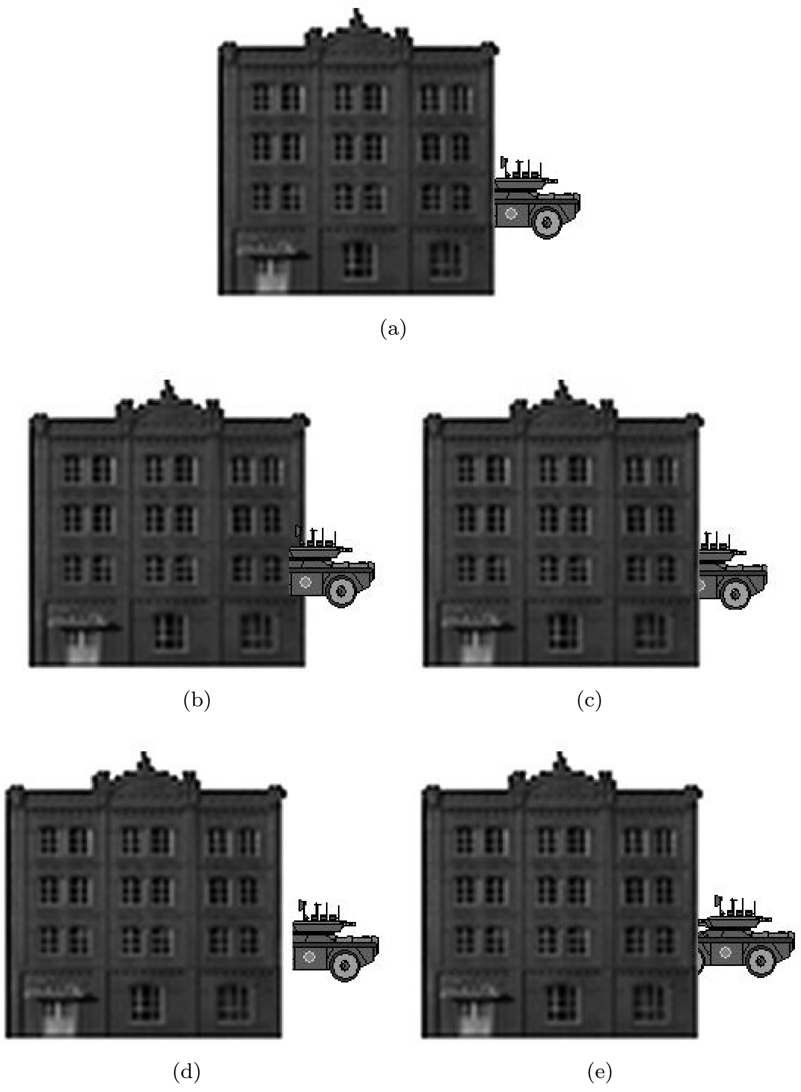


FIGURE 2. A simple example of a virtual vehicle behind a building for illustrating the limitations of current techniques. (a) A virtual vehicle behind a virtual building. (b, d) Small errors in registration cause visual inconsistencies because current techniques cannot tolerate any small errors in registration. (c, e) Small errors in registration do not cause visual inconsistencies because our technique can tolerate small errors in registration.

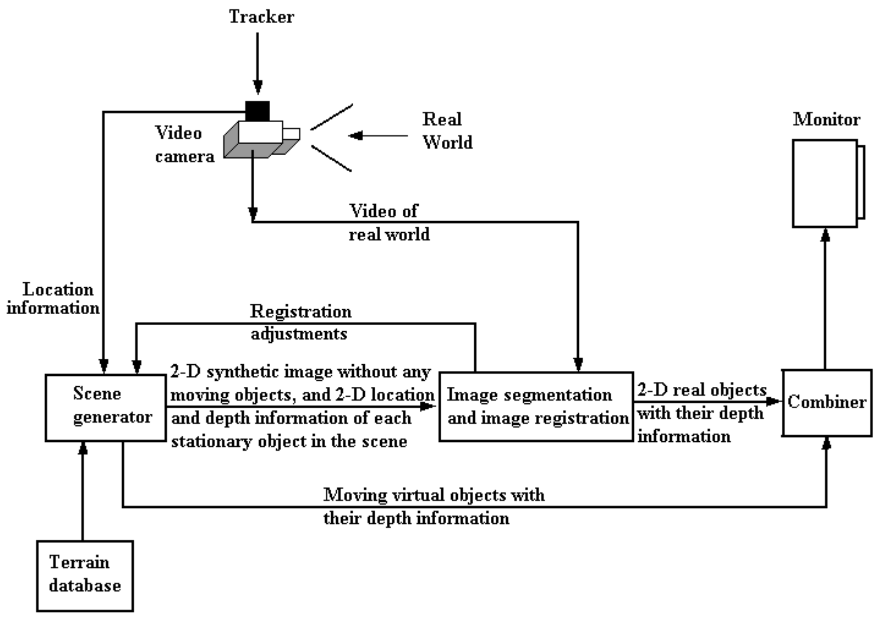

Tracker
Video camera
Real World
Monitor
Video of real world
Location information
Registration adjustments
Scene generator
2-D synthetic image without any moving objects, and 2-D location and depth information of each stationary object in the scene
Image segmentation and image registration
2-D real objects with their depth information
Combiner
Moving virtual objects with their depth information
Terrain database

Figure 3. System architecture.

2.1. System architecture

A schematic representation of the system we have developed is shown in Figure 3. The simplest method of capture of the **Live Environment** is through a video camera providing color RS-170 video, providing a two-dimensional representation of the scene. Camera pointing angles and camera zoom information are available to a PC so that we can determine the aim-point, and generate a synthetic image that is equivalent to the live image. The TB format of terrain we have used is the SAF (Semi-Automated Forces) [11,16] environment provided *via* topological terrain formats. Another approach would be to represent the virtual environment by visual databases used in stealth environment applications. For the virtual representation, we have chosen OpenGL which renders three-dimensional virtual scenes in two dimensions. As part of the rendering process in OpenGL, the depth (Z-buffer) of each object in the scene is determined. This depth is then used to calculate which objects are visible from the current aim-point.

The block diagram in Figure 3 outlines the flow of data in the system. A camera provides the live terrain image. A tracker is attached to the camera to provide the location and the direction of the camera. The scene generator uses this location information to generate the 2-D synthetic image, and 2-D location and depth information of each stationary object in the field of view. Our image segmentation algorithm uses this information to segment the live image into 2-D real stationary objects. Since we cannot obtain depth information from the live image in real time, we assume that the depth of each 2-D real stationary object is identical to the depth of the corresponding virtual stationary object. The image registration algorithm uses the locations and the sizes of the virtual and real stationary objects to adjust the viewpoint of the virtual scene. The scene generator also generates the moving objects with their depth information. The combiner then inserts these moving objects based on their depth information between real stationary objects.

At the core of the system we have developed is an algorithm to insert synthetic moving objects into live images in real time. The essence of this approach is that it is purely computationally based, and that it does not make use of any direct video manipulation. A primary concern is the proper placement of the virtual objects in front of, or behind, live objects. Thus the realistic representation of the inserted objects is tied to both the appropriate occlusions and the shapes and sizes of inserted objects. Depth information from the virtual scene is used, so that there is no need for additional instrumentation.

The decomposition of a live image into objects is shown in Figure 4. Once the object representation of the live scene is determined, the insertion of a synthetic moving-object will be based on the distance at which this object should be placed. Figures 5–7 illustrate the insertion of synthetic-target objects into the live scene. Figure 8 shows a test of our system using a real world scene.

3. Autonomous behavior of injected objects

The behavior of injected artificial entities can be as important as their appearance in a Visual Simulation. This is especially important in the context of simulations designed for training personnel or evaluating a "what if ..." situation. In such simulations, the behavior of agents will have an important effect on the final outcome in the form of "acquired training experience". Unrealistic agent behavior, *e.g.*, in the form of very limited or even extremely advanced intelligence may lead to poor performance of the trainees in a real-life situation.

Agent behavior in a sophisticated simulated environment can be very complex and may involve many entities. Intelligence can be employed at very different levels. A very simple example will be an agent that has to go from one position to another trying to minimize travel time. A very complex example of intelligent behavior can include the decision to cancel the mission of a group of entities and relocating them as a backup for another group. While the first problem can be easily solved by a single autonomous entity, the second will involve some authority that can make a higher-level decision based on information feedback from the lower-level autonomous agents.

Multi-Agent systems are a natural way of dealing with problems of distributed nature. Such problems exist in a diversity of areas like military training, games and entertainment industry, management, transportation, information retrieval and many others. The classical approach to AI, until now, has been unable to provide a feasible approach for solving problems of this nature. The need for such tools has lead to the "alternative" approach of behavior-based systems, popularized by the works of Brooks [6] and Arkin [1]. This approach takes simple behavior

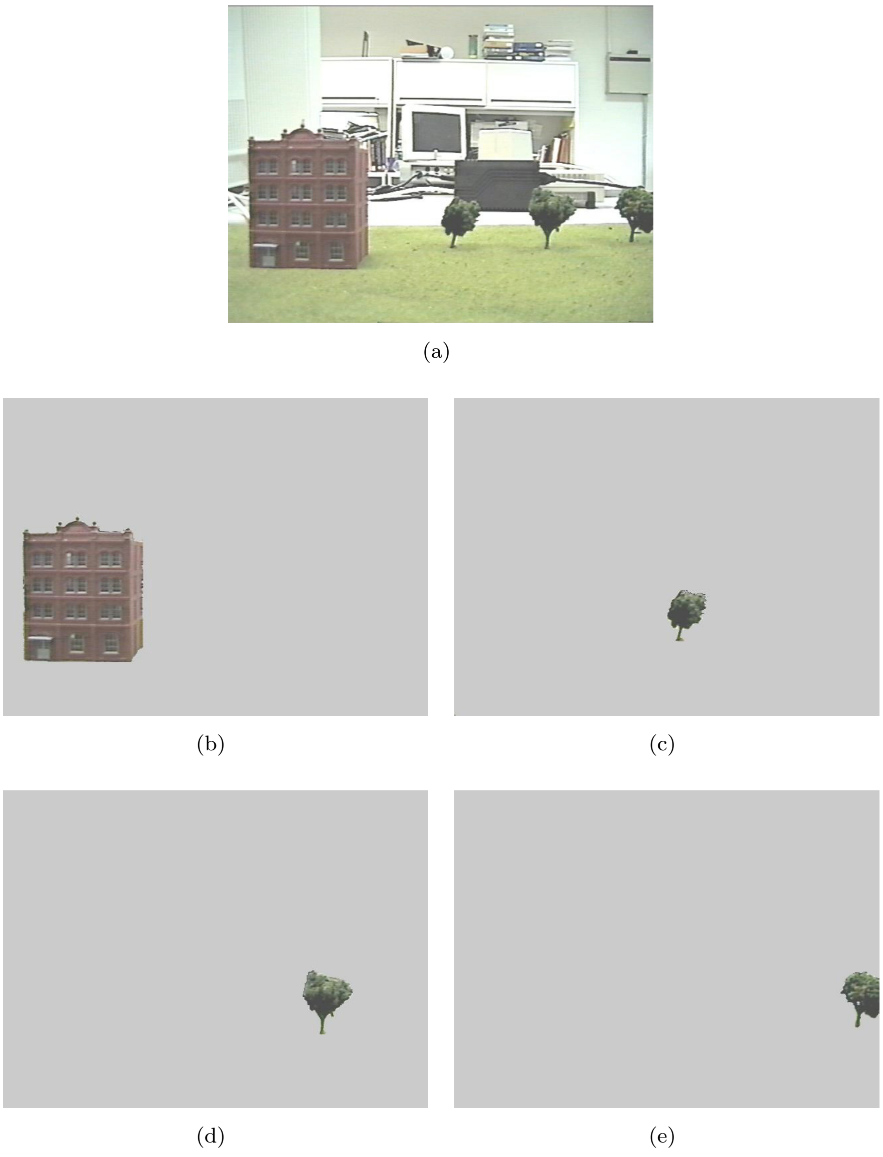
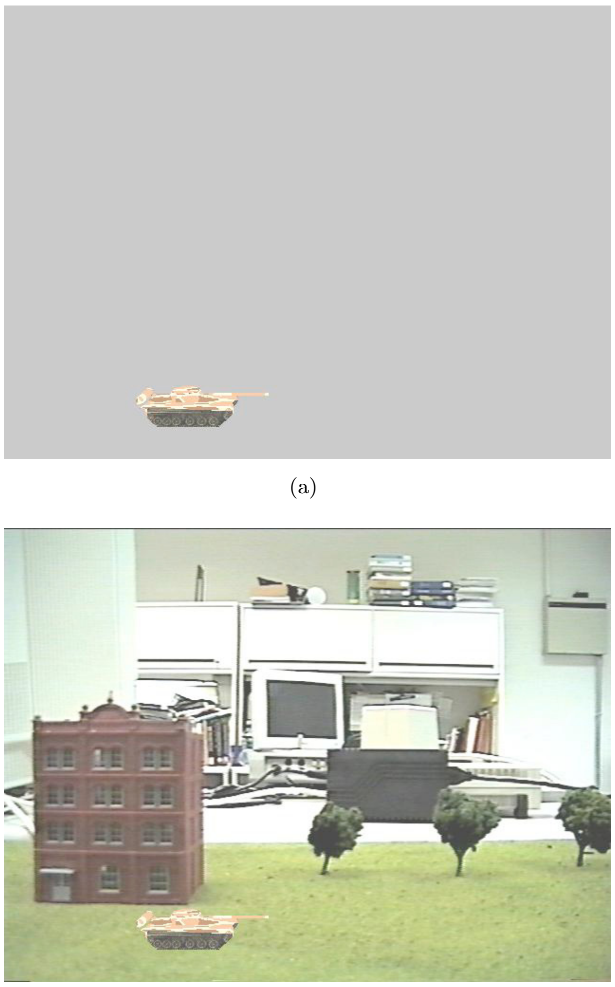


FIGURE 4. The decomposition of the live image into objects. (a) Table-top camera scene. (b) The building. (c) The left tree. (d) The middle tree. (e) The right tree.



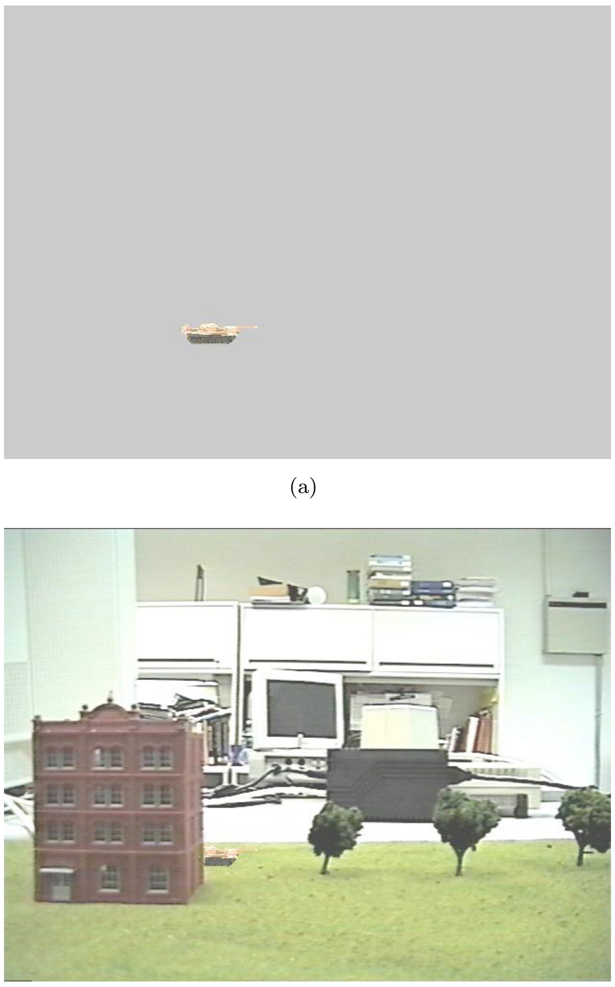
(a)

(b)

FIGURE 5. (a) A synthetic moving object, and the result (b) of inserting it into the live scene.

patterns as basic building blocks and tries to implement and understand intelligent behavior through the construction of artificial life systems. Its inspiration comes from the way intelligent behavior emerges in natural systems studied by Biology and Sociology. Good discussions on the development of behavior-based AI can be found in [7, 37] and an extensive treatment of the subject is given in [2].

Agents have to dynamically deal with an environment that they discover in the course of the simulation, and this must occur with decision algorithms which act rapidly within the realistic time frame of the realistically visual simulation,

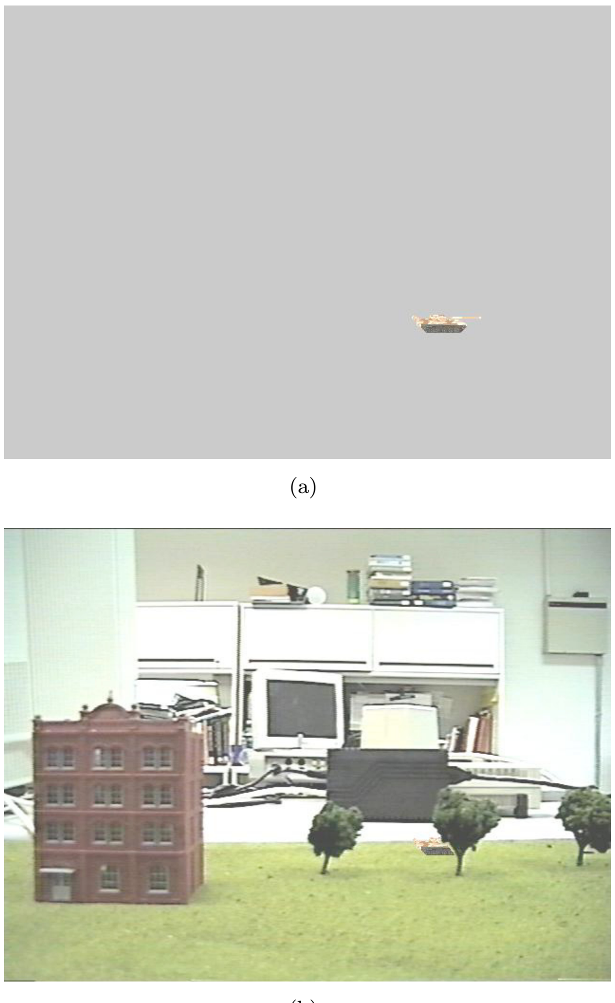


(a)

(b)

FIGURE 6. (a) A synthetic moving-object, and the result (b) of inserting it into the live scene.

which excludes the use of slow deductive or theorem proving techniques. Agents must be able to learn, coordinate and collaborate with each other in real time. Reinforcement Learning emerges as one of the "natural ways" of dealing with the dynamism and uncertainty of the environment. The complexity of the environment and the strict timing constraints however make the learning task extremely difficult. Even simple multi-agent systems consisting of only a few agents within a trivial environment can have prohibitively expensive computational requirements related to learning [31, 32, 38].



(a)

(b)

FIGURE 7. (a) A synthetic moving-object, and the result (b) of inserting it into the live scene.

Behavior-based systems have been more successful at dealing with such problems. Biology-inspired models of group behavior such as Reynold's "boids" [35] and approaches based on potential fields [34] address group behavior at a reasonable cost. Because of their performance and ability to scale better, they have been widely employed in technology-driven fields such as the computer-games industry [33, 36]. One of the main problems of behavior-based systems is that their

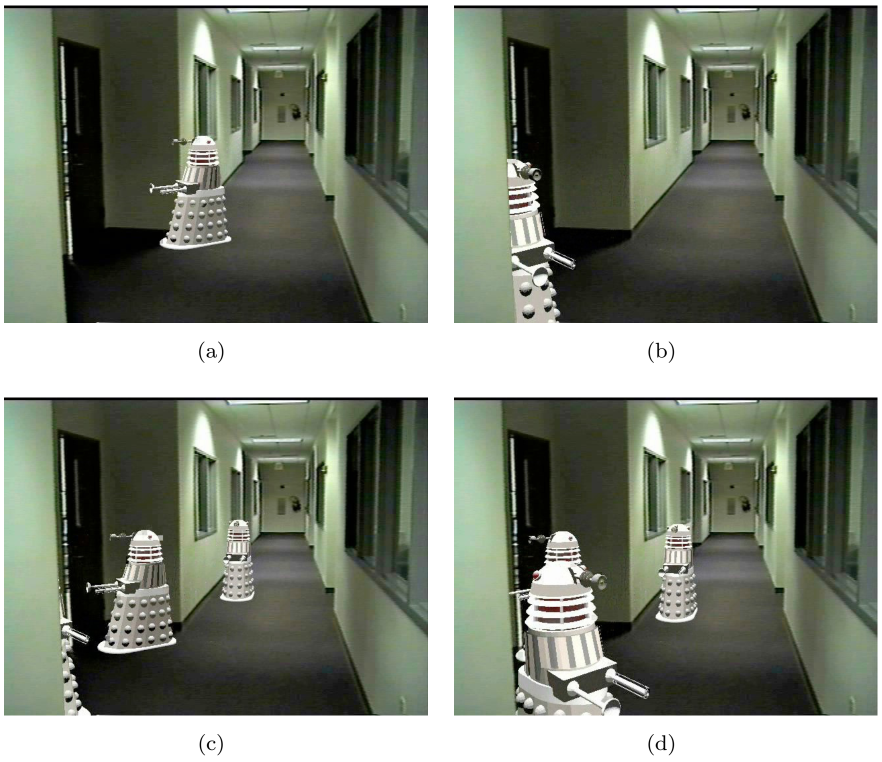


Figure 8. Some frames from an augmented video sequence of a corridor containing a swarm of synthetic robots, which shows how we use actual video from the real world together with synthetic mobile objects. (a) Frame No. 595. (b) Frame No. 620. (c) Frame No. 830. (d) Frame No. 835.

constituents can be very easily caught in local-minima. Multi-Agent Reinforcement Learning in the behavior domain [4, 29] is an actively explored approach to solve these problems in a robust way.

Our agent model is designed with the primary application area being military visual simulations. The particular problem explored in this paper is goal-based navigation of a group of autonomous entities in a dangerous terrain. The design of the agent model is based on the assumption that agents will perform "outdoor" missions in a terrain which is relatively sparse with respect to obstacles and enemies. It is not very suitable for "indoor" missions like moving inside a building or a labyrinth, where a more specialized approach will be more successful. A "mission" in our model is defined as the problem of going from some position A to some other position B avoiding being hit by an enemy or crashing into the natural and artificial obstacles present in the terrain. The success of the mission is measured by the amount of time necessary for completing the goal and the survival rate of the agents.

Our approach is based on a hierarchical modular representation of agent behavior. This method allows for de-coupling the task of group navigation into simpler self-contained sub-problems which are easier to implement in a system having computational constraints due to interaction with real-life entities. Three basic modules that we call the *navigation* module, *grouping* module and *imitation* module are used to control the behaviour of the mobile entities. The Navigation Module is responsible for leading a single agent from a source location to a destination location, avoiding danger and obstacles. The Grouping Module is responsible for keeping a group of agents together in particular formations throughout the mission. The Imitation Module is modeling the case when an inexperienced agent will try to mimic the behavior of the most successful agents in the group and thus increase its chances of success, and the decisions of these modules are combined at a higher level by the *Coordinator Module*.

The Navigational Module generates moves based on a quantized representation of the simulated environment in the form of a grid. Terrain properties are assumed to be uniform within each grid cell for the purpose of learning and storing information about the terrain. Each cell in the grid represents a position and an "agent action" is defined as the decision to move from a grid cell to one of the eight neighboring cells. A succession of such actions will result of a completion of a mission. The agents can also access terrain-specific information about features and obstacles of natural (trees, etc.), and artificial origin (buildings, roads, etc.) and also presence of other (possibly hostile) agents. The interaction between an enemy (a hostile agent) and an agent is modeled by an associated risk. This risk is expressed as a probability of being shot (for an agent) at a position, if the position is in the firing range of an enemy. The goal of the agent is to minimize a "goal" function G (for instance, the estimated time of a safe transit to the destination). The Navigational Module of an agent has a so-called "cognitive map" which is a collection of latest and smoothed rewards for each decision taken at each visited grid cell. The decision-making element of a Navigation Module is a fully-connected RNN network, where each neuron represents a possible alternate decision. Its training is performed by reinforcing the weights of each neuron, depending on the difference between the observed and historical rewards. A detailed description of the network and the learning process can be found in [26, 27]. By using previously acquired information and current sensory input, an agent can start with near-optimal estimates of the rewards and skip an otherwise prohibitively-long learning session and focus on adapting to the dynamic changes in the environment.

The grouping behavior module is based on the idea of social potential fields [34] which is a simple distributed-control approach inspired by the attractive and repulsive forces between charged particle in physics. Although this method has been used in a broader domain (including path-planning), we restrict its usage only to model grouping behavior for which it is particularly well suited. Using potential fields methods for other purposes like generalized navigation and obstacle avoidance requires dealing with local minima problems and difficult to design force-configurations that can easily nullify the simplicity gained by using the method in the first place. These behaviors are very similar and can be used to get the effect of the *collision avoidance* and *flock centering* rules as described by Reynolds [35]. By setting up a two-way mesh of forces between a number of agents, for example, a spatially localized group can be created that will try to stay together. Another simple example is a one-way mesh of forces from the leader of the group to the other members, suggesting that they should stay close and follow if necessary the leader, without having any effect on his decision making.

The purpose of imitation is to efficiently take advantage of experience without going through the trouble of actually acquiring it – that is, it has a much lower computational cost, compared to the other methods. The imitation module proposes a decision which is a weighted sum of the navigational decisions of some of the members of the agent group. The weights can change over time, in order to reflect the group members which are currently observable or known to be experienced. The *velocity matching* flock behavior described in the work of Reynolds [35] which he defines as "attempt to match velocity with nearby flocks" is a very similar idea.

Figure 8 shows a few frames from a simulation which uses these techniques, in conjunction with the visual augmented reality system we have described, to investigate the behaviour of robotic swarms operating within an existing setting (a corridor linking several university laboratories).

4. Conclusions

Modern discrete event simulators often require the representation of complex autonomous behaviors within a visually realistic setting. They often use a graphical interface both as an input and as an output medium to simplify and enrich the user's interaction with the simulation both before, during and after the simulation runs. Many simulation tools also provide an animated graphical interface which offers a real-time visual description of a simulation in real time.

A useful and very significant leap forward in simulation technology is to be able to evaluate synthetic simulated conditions in realistic settings. The idea here is to ask questions about "what would happen if ..." in the context of a real environment and actual events. This challenge is the focus of the work addressed in this paper where we mix simulation with reality in real time, in order to examine how novel simulated conditions can actually interact with a real system's operation. This interaction can go in both directions: the course of the real world can be modified by virtual entities, and the virtual objects are constrained to operate in the real world.

In this paper we have discussed the conceptual issues which arise in this key area of visual simulation, and we have presented some design principles and a practical implementation. Key issues we covered in this paper include a new method for injecting moving synthetic objects in real-time into real world video based on terrain databases, graphics rendering and image segmentation, and a novel approach to automatically control the motion of synthetic agents within the realistic live scene.

References

[1] R.C. Arkin, Motor Schema-Based Mobile Robot Navigation. *Int. J. Robot. Res.* **8** (1989) 92-112.

[2] R.C. Arkin, *Behavior-Based Robotics.* The MIT Press, Cambridge, Massachusetts (1998).

[3] M. Bajura and U. Neumann, Dynamic registration correction in video-based reality systems. *IEEE Comput. Graph. Appl.* **15** (1995) 52-60.

[4] T. Balch, *Behavioral Diversity in Learning Robot Teams.* Ph.D. Thesis, Georgia Institute of Technology (1998).

[5] E. Gelenbe, Reseaux neuronaux aleatoires stables. *C.R. Acad. Sci.* II **310** (1990) 177-180.

[6] R. Brooks, A robust layered control system for a mobile robot. *IEEE J. Robot. Autom.* **RA-2** 14-23 (1986).

[7] R. Brooks, *Cambrian Intelligence: The Early History of The New AI.* The MIT Press, Cambridge, MA (1999).

[8] C. Cramer, E. Gelenbe and H. Bakircioglu, Low bit rate video compression with neural networks and temporal subsampling. *Proc. IEEE* **84** (1996) 1529-1543.

[9] C. Cramer and E. Gelenbe, Video quality and traffic QoS in learning-based subsampled and receiver-interpolated video sequences. *IEEE J. Selected Areas in Communications* **18** (2000) 150–167.

[10] Y. Feng and E. Gelenbe, Adaptive object tracking and video compression. *Netw. Inform. Syst. J.* **1** (1999) 371-400.

[11] B. Foss, E. Gelenbe, K. Hussain, N. Lobo and H. Bahr, Simulation driven virtual objects in real scenes. *Proc. ITSEC 2000*, Orlando, FL (Nov. 2000).

[12] E. Gelenbe, Reseaux stochastiques ouverts avec clients negatifs et positifs, et reseaux neuronaux. *C.R. Acad. Sci. Paris* II **309** (1989) 979-982.

[13] E. Gelenbe, Random neural networks with positive and negative signals and product form solution. *Neural Comput.* **1** (1989) 502-510.

[14] E. Gelenbe, Stable random neural networks. *Neural Comput.* **2** (1990) 239-247.

[15] E. Gelenbe, Learning in the recurrent random network. *Neural Comput.* **5** (1993) 154-164.

[16] E. Gelenbe, Modeling CGF with learning stochastic finite-state machines. *Proc. 8th Conference on Computer Generated Forces*, Orlando, May 11–13, 113-116.

[17] E. Gelenbe, Simulation with goal-oriented agents. in EUROSIM 2001. Delft University of Technology Delft, Netherlands, 26–30 June (2001).

[18] E. Gelenbe, Applications of spiked recurrent stochastic networks in *13th International Conference on Artificial Neural Networks & 10th International Conference on Neural Information Processing*, 26–29 June (2003).

[19] E. Gelenbe, Spiked random neural networks, product forms, learning and approximation in *Conference on Analytical and Stochastic Modeling Techniques and Applications, European Simulation Multi-conference*, Nottingham, UK, 9–11 June (2003).

[20] E. Gelenbe, C. Cramer, M. Sungur and P. Gelenbe, Traffic and video quality in adaptive neural compression. *Multimedia Systems* **4** (1996) 357-369.

[21] E. Gelenbe, T. Feng and K.R.R. Krishnan, Neural network methods for volumetric magnetic resonance imaging of the human brain. *Proc. IEEE* **84** (1996) 1488-1496.

[22] E. Gelenbe and J.M. Fourneau, Random neural networks with multiple classes of signals. *Neural Comput.* **11** (1999) 953-963.

[23] E. Gelenbe and K. Hussain, Learning in the multiple class random neural network. *IEEE Trans. on Neural Networks* **13** (2002) 1257-1267.

[24] E. Gelenbe, K. Hussain and V. Kaptan, Realistic simulation of cooperating robots, in *Proc. CTS'03 (International Symposium on Collaborative Technologies and Systems)*, WMC'03 Society for Computer Simulation, Orlando, 19–23 January (2003), 151-156.

[25] E. Gelenbe, V. Kaptan and K. Hussain, Simulating Autonomous Agents in Augmented Reality. Submitted for publication.

[26] E. Gelenbe, E. Şeref and Z. Xu, Discrete event simulation using goal oriented learning agents, *AI, Simulation & Planning in High Autonomy Systems*, SCS, Tucson, Arizona, March 6–8 (2000).

[27] E. Gelenbe, E. Şeref and Z. Xu, Simulation with learning agents. *Proc. IEEE* **89** (2001) 148-157.

[28] S.W. Lawson, Augmented reality for underground pipe inspection and maintenance, in *SPIE Conference on Telemanipulator and Telepresence Technologies*, Boston, Massachusetts, **3524** (1998) 98-104.

[29] M. Mataric, Reinforcement learning in the multi-robot domain. *Autonomous Robots* **4** (1997) 73-83.

[30] J.P. Mellor, *Enhanced reality visualization in a surgical environment.* M.S. Thesis, Department of Electrical Engineering, MIT (January 1995).

[31] N. Ono and K. Fukumoto, Multi-agent reinforcement learning: A modular approach in *Proc. of the 2nd Int. Conf. on Multi-Agent Systems.* AAAI Press (1996) 252-258.

[32] N. Ono and K. Fukumoto, A modular approach to multi-agent reinforcement learning, edited by Gerhard Weiss in Springer-Verlag, *Distributed Artificial Intelligence Meets Machine Learning*, (1997) 167.

[33] D.C. Pottinger, Implementing Coordinated Movement. *Game Developer Magazine* (1999).

[34] J.H. Reif and H. Wang, Social Potential Fields: A Distributed Behavioral Control for Autonomous Robots, in *International Workshop on Algorithmic Foundations of Robotics (WAFR)*, edited by A.K. Peters, Wellesley, MA (1998) 431-459.

[35] C.W. Reynolds, Flocks, Herds, and Schools: A Distributed Behavioral Model. *Comput. Graph.* **21** (1987) 25–34.

[36] C.W. Reynolds, Steering Behaviors for Autonomous Characters. *Game Developers Conference* (1999).

[37] L. Steels, The Artificial Life Roots of Artificial Intelligence. *Artificial Life* **1** (1993) 75-110.

[38] M. Tan, Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents, in *International Conference on Machine Learning* (1993) 330-337.