Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 00 (2012) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

Virtual Worlds for Serious Applications (VS-GAMES'12)

Semantic crowds: reusable population for virtual worlds

Nick Kraayenbrink[a], Jassin Kessing[a], Tim Tutenel[a], Gerwin de Haan[a], Fernando Marson[b], Soraia R Musse[b], Rafael Bidarra[a,*]

[a]*Computer Graphics and Visualisation Group, Delft University of Technology, Delft, The Netherlands*
[b]*Virtual Humans Simulation Lab, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil*

Abstract

Recent advances in crowd simulation techniques have led to realistic agent and group behavior through elaborate behavioral models, complex motion planning algorithms and impressive physics systems. As many crowd simulation solutions typically target only specific types of environment and scenario, a variety of special-purpose methods and systems has emerged that are hard to re-configure and re-use in other contexts. Solving this situation demands a higher-level approach that takes re-use and configuration of crowds as a priority, for adequate application in a broad variety of scenarios, virtual environments and interaction with the entities present in that environment. In this article we propose *semantic crowds*, a novel approach that allows one to re-use the same crowds for virtually any environment, and have them use the objects available in it in a meaningful manner, without any modification. To have the agents autonomously interact within any virtual world, we minimize in them the information relative to what objects do and how to use them. Instead, that information is stored in the objects themselves, which the agents can then query, based on what they plausibly want to achieve. To facilitate creating such crowds, we developed an interactive crowd editor that provides high-level editing parameters for defining crowd templates. We illustrate the flexibility of semantic crowds by means of two cases, in which we let the same crowd populate quite differently configured airport terminal environments. These examples also highlight that this modular approach easily combines with your custom implementations of agent behavior model and/or motion planner.

Keywords: crowd specification, reusable crowds, semantic virtual worlds

1. Introduction

Real-time crowd simulation is becoming increasingly important for a large variety of applications, such as games and simulation systems for training and education. For example, entertainment and serious games can expect a fair amount of criticism, if they feature rather unpopulated and empty cities and other environments. Recent advances in crowd simulation techniques have enabled more and more realistic agent and group behavior

*Corresponding author. E-mail address: r.bidarra@tudelft.nl deploying, among other things, elaborate behavioral models, complex motion planning algorithms and impressive physics systems.

However, these improvements come at a price: most crowd simulation solutions are either too generic and high-level, or they are pretty much 'hard-wired' and customized, typically targeting only a few specific types of environment and scenario. The former likely leads to repetitive, 'canned behavior', the latter to an ad-hoc, specialized crowd, only suited for a very narrow application domain. In both cases, we are faced with a large variety of special-purpose methods and systems that are hard to re-configure and re-use in other contexts.

Avoiding these drawbacks requires a totally different approach, one that primarily supports and promotes the specification, configuration and re-use of crowds. In this article we propose one such approach, called *semantic crowds*, that allows one to specify and re-use the same crowds for virtually any environment, and have them use whatever objects are available in it in a meaningful manner, without any modification.

We argue that, in order to successfully apply such specifications, or *crowd profiles*, in a broad variety of scenarios and virtual environments, the latter have to contain more than just pure geometry: they have to be extended with *semantics*. In the fields of linguistics, computer science and psychology, semantics is the study of meaning in communication. When focusing on virtual environments, we call semantics 'all information conveying the meaning of a virtual world and its entities' [1]. In this definition, entities encompass everything that can exist inside a world. Semantic virtual worlds, thus, consist of entities that 'know' not only about their shape and materials, but also about their attributes, roles, functions, services, etc. Such entities have the potential to strongly improve the quality and variety of interactions with an avatar, be it of a player or of any agent from a crowd [2].

In order to have the agents in a semantic crowd plausibly and autonomously interact within any virtual world, we minimize the information in the agents relative to what objects do and how to use them. Instead, that information is stored in the objects themselves, which the agents can then purposefully query, based on what they want to achieve.

For the specification of crowds and agent behavior, we integrate several concepts, results and methods known from the literature. The novel contribution of our work lies mainly in that the same crowd, once specified, can be applied to whatever different virtual environments, while its agents always exhibit plausible yet very different behaviors without having to be 're-wired' themselves. In other words, we are making such crowds effectively reusable.

The paper is structured as follows. We first survey previous work related to the specification of both virtual crowds and environments (Section 2). Next we elaborate on the essential aspects of semantics, particularly on its contribution to crowd and environment definition (Section 3). We then introduce the main concepts of the semantic crowd model, both in its demographics and in its agents (Section 4), and describe the main features of our prototype framework, including an interactive crowd editor that provides high-level editing parameters for defining crowd profiles (Section 5). Finally, we illustrate the potential and flexibility of semantic crowds by means of two demonstration cases (Section 6) and draw some conclusion (Section 7).

2. Related Work

This section surveys a selection of previous work regarding the representation and generation of virtual environments and crowds.

2.1. Virtual Environments

Beyond geometric data, virtual environments may also contain information about objects, which actions these can perform or how they can be used by agents. A Virtual Environment (VE) presenting this kind of information is normally referenced as an Intelligent Virtual Environment [3], Informed Environment [4] or Semantic Virtual Environment [5].

Thomas and Donikian [6] propose a model of virtual environments using structures suitable for behavioral animations. Using this knowledge, autonomous virtual actors can behave like pedestrians or car drivers in an urban environment.

Through the concept of synoptic objects [7], Badawi and Donikian describe an informed environment using objects which contain a synopsis of interactions that they can be subjected to. Using a set of seven basic actions, the objects can describe the interaction process to any agent using them.

Smart objects [8] were a successful proposal for adding semantics to virtual objects, dealing with many of the possible user interactions in a VE. They were primarily devised for manipulation, animation, and planning purposes.

Gutierrez, Vexo and Thalmann [9] present an object representation based on the semantics and functionality of interactive digital items within a VE. Each object participating in a VE application in not only a 3D shape, but a dynamic entity with multiple visual representations and functionalities. In this way is possible dynamically scaling and adapting the object's geometry and functions to different scenarios.

Research in artificial intelligence proposed the notion of ontologies, due to the lack of shareable and reusable knowledge bases [10]. Ontologies define the meaning of objects and the relations between them. Using this concept to define a world knowledge base, Grimaldo et al. [11] propose a semantic-based framework for simulation of groups of intelligent agents. Agents can use the provided information to enhance both agent–object and agent–agent interactions. An object taxonomy is used to classify the interactive objects and to get their properties. Ontologies are also used to define social relations among agents in order to display socially acceptable decisions.

A semantic model for representing multi-layered complex environments is presented by Jiang et al. [12] and is composed of three different levels: a geometric level, a semantic level, and an application level. The geometric level contains a 3D model of the environment that is used for visualization and to extract semantic information that will feed the next semantic level. This semantic level comprehends a structure map, a topologic map and a height map which are used to identify or query semantic information of the environment. The final layer (application level) is responsible for providing efficient interaction between pedestrians and the environment. The crowd model used in their work is a modified version of the original model proposed by Treuille et al. [13].

2.2. *Crowds*

Virtual groups have been studied since the early days of behavioral animation. There are two main approaches used in the literature: agent-based models and force-based models. The first approach is based on modeling of virtual agents, which interact among themselves and have some level of autonomy and individuality. The second approach –force-based models– provides more global control and handles high density crowds. These methods normally yield crowd simulations that resemble particles rather than human animation. In this subsection, we survey existing approaches for crowd models.

Two seminal papers are examples of agents-based model: Reynolds [14] simulated flocks of bird-like entities, or boids, obtaining realistic animation by using only simple local rules; Tu and Terzopoulos [15] created groups of artificial fishes endowed with synthetic vision and environmental perception, which control their behavior. In both papers, only small groups were simulated, and high density crowds were not treated. Another example of agents-based models was presented by Musse and Thalmann [16], who proposed an approach with hierarchically structured crowds having different levels of autonomy. In their model, the behavior is based on a set of rules dealing with the information contained in groups of individuals, e.g. individual and group knowledge about the world, individual and group states as well as their intentions. Ulicny and Thalmann [17] proposed a model for crowd simulation based on combination of rules and Finite State Machines for controlling agents' behaviors in a multi-layer approach. The model proposed by Farenc et al. [18] describes the coordination between smart objects, intelligent environments and the virtual crowd. This work describes crowds of virtual people which were controlled by the objects in order to act, and could read the environment in order to evolve in the simulation. In this case, the virtual agents were less autonomous and more controlled by the environment. Not focused on crowds, but on groups of few virtual agents, the model proposed by Abaci et al. [19] describe an extended version of smart objects for AI and planning purposes.

The second class of crowds in literature is concerned with force-based models. Bouvier and collaborators [20] have studied crowd movements with adapted particle systems. In their work, the motion of people is modeled with charges and decision fields, which were based on the interactions between electric charges and electric fields. Helbing and collaborators [21, 22] presented a model to simulate groups of people in panic situations.

Helbing's model is physically-based, and crowd movement is determined by attraction and repulsion forces between simulated agents and the environment. Both models are examples of force based methods that deal with high density crowds, where virtual agents are homogeneously treated as particles. Braun et al. [23] extended this model by adding individualities to agents and including the concept of groups, focusing on panic situations. Despite the interesting results, this latter model does not present facilities to provide individual control. In addition, physically-based models as proposed by Bouvier et al. [20], Helbing and Molnar [21] lack simplicity and robustness, i.e., any changes in resulting behavior should be addressed by changes in the equation that represents the global control. Treuille et al. [13] proposed a real-time crowd model based on continuum dynamics. In their model, a dynamic potential field integrates global navigation with moving obstacles, by modeling the spatial motion of agents as a function of crowd density. This model presents very interesting results, but the model lacks local control and simplicity. In addition, changes in the model are hard to achieve because of its complexity.

Finally, some hybrid methods describe models to combine local and global control, e.g. Pelechano et al. [24] described a crowd model by applying a combination of psychological and geometrical rules, with social and physical forces. This model is focused on panic situations, and many parameters should be calibrated in order to exhibit expected behaviors.

In summary, some of the simulation methods presented in literature can be classified as agents-based models. While this popular approach brings some behavioral advantages in obtained results, it lacks of control flexibility in high dense crowds. On the other hand, the model described in [13] is one example of a force-based model. It presents interesting assumptions dealing with individuals and groups with common goals. However, the downside is the complexity of the method and the lack of individual control.

3. Semantics

In this section, we elaborate on the semantics required for both virtual environments (3.1) and agents (3.2).

3.1. Semantics for Virtual Environments

Virtual worlds are populated with objects that are described by various *properties*, by an *appearance* and, increasingly often, by some *physics* as well. In addition, an object can exhibit some basic behavior, usually defined by scripts that, for example, can prescribe how to move, what animation to trigger, or how to interact with other objects. We developed a semantic model that provides a shared knowledge base among all these components (presentation, physics and behavior).

We define a *semantic game world* as 'a virtual environment that is populated with entities (either objects or agents) that are enriched with semantics' [25]. Because of this semantics, all information in the world can be kept consistent. Not only is this valid for the external appearance of an entity, but also for its internal behavior, and its relations with other entities. When more entities behave like in real life, both players and agents can achieve their goals in multiple ways. For example, in order to satisfy their hunger, agents might eat a snack, buy food in a nearby shop, or order a meal in a restaurant, assuming these entities were specified in a proper way.

The main concepts in our semantic model are *entities* (*physical objects* or *spaces*), of which characteristics are expressed in *attributes*. Physical objects consist of a particular type of *matter* on which attributes are defined as well. Many relationships can be expressed between such entities, and be used, for example, to automatically generate floor plans [26], room layouts [27] and consistent buildings [28].

The generic behavior of entities in our model is expressed through the concept of *services*, defining 'the capacity of entities to perform particular actions' [2]. Basic components of an action are its *requirements* and its *effects*. The effects determine what the action brings about, while the requirements determine when the action may be performed. For example, when someone inserts a coin into a vending machine, it will return a snack, which can be eaten to reduce the level of hunger.

3.2. Semantics for Agents

In order to simulate agents in a semantic game world, the concept of *desires* has been introduced in our semantic model. A desire describes the intent of an entity with regard to its state, and can be either relative or absolute. Absolute desires describe the state or the range of states the entity wishes to be in, and the set of states that satisfies these desires will not vary over time. For example, an agent may have the desire to have at least 10 coins. In contrast, relative desires describe changes that the entity wishes to apply to its state, and thus the set of states that satisfies these desires can change when the entity's state changes. For example, for an agent to 'obtain another hat', it does not matter how many hats the agent already has, only by having one more will the desire be fulfilled.

By querying their (known part of the) world, agents will be able to find actions whose effects have an influence on their current desires. From each action requirement, a new desire can be generated to satisfy it, for which actions can then be found that may help satisfy that desire. Actions may also have a reaction as one of its effects (the attempted start of another action), and as such agents may also include actions performed by other entities in their search.

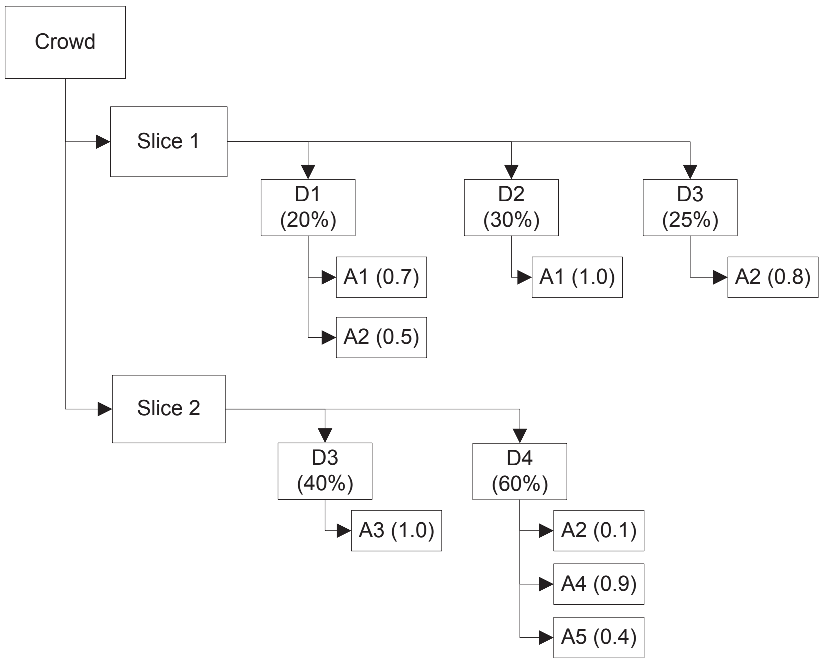


Fig. 1: Example of the structure of a crowd using our agent model. D# represent Demographics, and A# Ambitions.

4. Crowds

The proposed semantic crowds approach is subdivided into two major parts: the crowd and the agent. The crowd model contains the global composition of a crowd in different demographics and the ratio in which they are present. For each of these demographics, a semantic agent model describes their specific behavior. The next sections will elaborate on both the crowd model and the semantic agents.

4.1. Crowd Model

The semantic crowd model consists of two independent components: the *crowd profile* and the *crowd socket*. The crowd profile defines all environment-independent aspects of the crowd. The crowd socket configures all aspects required to insert the crowd into a concrete environment.

4.1.1. *Crowd Profile*

In this section, we discuss the crowd profile of our model. We work from the bottom up, from one level above the agents themselves, up to the entire crowd. Note that with our definition of crowd, there can be more than one crowd acting in a single environment. We use the term *population* for the entire set of agents in an environment.

A demographic is a group of agents that have something in common, and it has often been used before in virtual crowd definitions, such as [29]. What they have in common depends on the used agent model. For example, the demographic 'heavy smokers' may contain agents that will often stop to smoke a cigarette. A demographic always contains a set of conditions, which constrain when an agent can belong to it, e.g. only when a market is busy enough pickpockets will appear. A demographic slice is a description of demographics that covers at most the entire crowd. One can think of this as the distribution of agents in demographics when a cross-section of the crowd is made. Each demographic is coupled with a percentage, e.g. shopping behaviors in a mall might be 30% shop-a-holics, 40% shoppers for groceries, 16% impulse buyers 1% thieves, and the other 12% passers-by. Since the demographics themselves have conditions, a demographic slice also contains an indicator on how to handle cases where the demographic is not enabled. If a demographic is disabled, it will either be treated as if it was unassigned space in the slice, or the other demographics will be 'enlarged' such that the slice covers the same percentage of the crowd.

A crowd is a collection of demographic slices, together with a 'default demographic'. Because the slices do not need to cover the entire crowd, it may happen that an agent is part of none of the demographics from the slices, in which case it will be part of that default demographic. A crowd also contains a distribution of agent attributes. Several attributes must always be present, such as age group (kid, adult or elder), gender and preferred movement speed. The used agent model may impose other required attributes. These distributions describe the general distribution of attributes across the crowd, and can be 'overridden' in each demographic. A schematic example of the full structure of a crowd profile can be seen in Figure 1.

4.1.2. *Crowd Socket*

In order to place a crowd in a concrete environment, some information on that environment is required, be it enriched with semantics or not. We now discuss the crowd socket component of our crowd model, and, in particular, how crowds as defined above can be inserted in concrete environments. The specifications listed below are the bare minimum required, and can be extended to facilitate extra features of the used agent model(s).

First and foremost, to insert a crowd into an environment it must be known which type of entity will represent agents (e.g. a generic human, a police officer or a car). Because a single environment may contain multiple types of crowds at once, the object type can be specified for each used crowd. Attributes of the 'controlled' entities may be set using the agent attribute distributions in the crowd and demographics.

It should also be specified where in the environment the agents will come (or *spawn*) from. Each type of spawn space must be specified individually and a desired spawn rate needs to be specified. The inverse type of space, the ones where the agents leave the environment, do not have to be specified in the crowd socket. Since not all exits need to be spaces, they have to be defined in the semantics by actions that remove the agent from the environment. The reason for this choice is that leaving should be an *action*, meaning that the agent must *want* to leave and simply being in the space designated as 'exit' does not necessarily mean that the agent intends to leave. Conditions may also be imposed on the spawn space to prevent agents from spawning. For example, agents might only spawn when the door is open, or while there are less than 500 agents in the environment.

4.2. *Semantic Agents*

This section discusses our agent model, which is the one that ultimately makes use of the semantics present in the environment. Aside from the global structure, including a specification of the accompanying type of demographic, we also discuss the global thought process of our agents.

4.2.1. *Agent Model*

Our agent model is a variant of the BDI (Beliefs, Desires, Intentions) agent model, first introduced by Bratman [30]. Agents have a set of goals that they wish to fulfil, each one containing one or more semantic desires (as introduced in Section 3.2). For each goal, they also have an urgency valuation function, or rather a way of determining how urgent each of the goals are. The combination of goal and urgency valuation function is what we call an ambition. Aside from assigning a numerical value to the urgency, the valuation function also determines if the ambition should be taken into consideration at all (for example, whatever happens, an agent should only buy a jacket once).

The set of ambitions in an agent are Desires in BDI, while Beliefs are implicitly present in the form of a world view. Intentions are represented by the ambition selected by the agent and the paths it found to satisfy it.

In the type of demographic specific for this agent model, a set of these ambitions is specified along with a probability. This probability indicates the chance an agent, randomly picked from all agents in the demographic, has that ambition. We also introduced two extra attributes for all agents: *determination* and *distractibility*. The higher the determination of an agent, the more likely it is that it will fulfil the ambition it selected before attempting to fulfil another one.

By letting a demographic be defined in a set of possible ambitions, we differentiate ourselves from works such as [29]. Instead of explicitly specifying where the agents will go at what times, we specify what the agents want to accomplish when certain conditions are met.

To make further use of ambitions, we also extended the crowd socket for our agents. Aside from the basic information, one can also specify actions that insert ambitions into agents, along with an insertion strength. Whenever those actions are successfully performed on an agent, the agent has a chance (determined by the insertion strength, as well as the agent's distractibility) to gain another ambition. Such a behavior of actions most closely resembles advertisements: they are designed to make agents want to do things they would never have thought of on their own.

4.2.2. Agent Thought Process

An overview of the 'thought process' of our agents can be seen in Figure 2. Note that this flow diagram only indicates the steps an agent will take when it is actually allowed to do something. If it is performing an action, or certain actions are being performed on the agent, this whole process is skipped and the agent just waits.

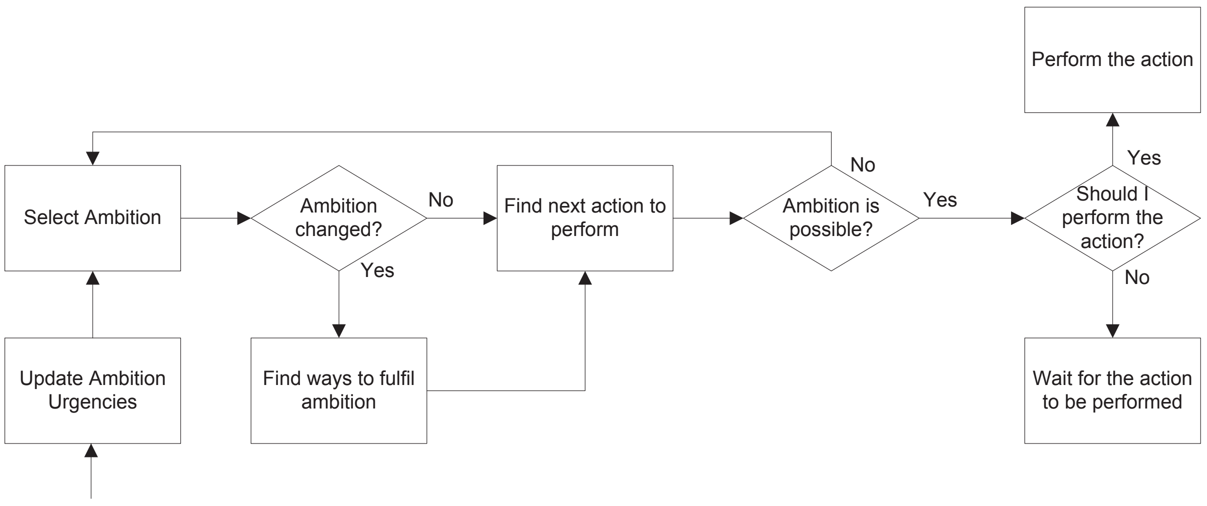

Perform the action
Yes
No
Select Ambition
Ambition changed?
No
Find next action to perform
Ambition is possible?
Yes
Should I perform the action?
Yes
No
Update Ambition Urgencies
Find ways to fulfil ambition
Wait for the action to be performed

Fig. 2: Overview of a single round in the thought process of a semantic agent.

Update Ambition Urgencies. First, the agent re-evaluates all its current ambitions, and determines which ambitions may be considered.

Select Ambition. Now the ambition with the highest urgency is selected. If the previously selected ambition has a high urgency as well, that ambition may be selected instead, depending on the agent's *determination*; the higher this determination, the higher the discrepancy in urgency values can be, without the agent selecting a different ambition.

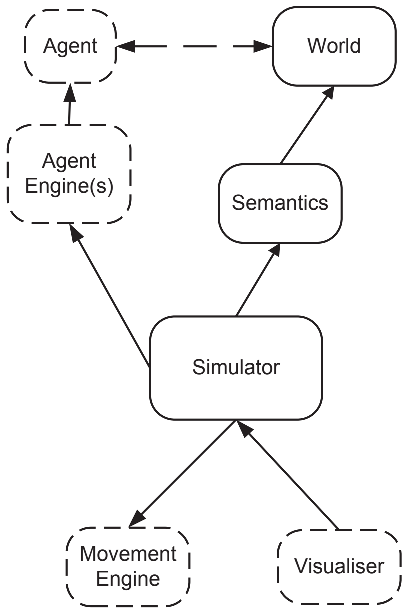


Fig. 3: Schematic overview of our framework.

Find Ways to Fulfil Ambition. If the selected ambition is different from the one the agent was trying to fulfil in the previous round, it needs to find out how to achieve its new ambition. The agent will query its world view for entities able to perform an action affecting one of the desires in its ambition. For each of those, desires are generated for its requirements, and so forth. This search only stops when all possible paths are found.

Find Next Action to Perform. Once both the ambition to achieve, and the ways to achieve it, are known, the agent can select the action to perform next. The action selection mechanism will not try to find the optimal way of achieving the ambition. Since the state of the environment can change while the agent is trying to fulfil its ambition, finding that optimal path is infeasible. Our agents use an action selection mechanism based on expected action cost. One could see this as the agents being lazy and short-sighted; they desire the least amount of effort, but will not take into account the amount of effort future actions might cost. A more graceful way of putting it is that agents opt for immediate gratification for instant feedback, instead of thinking of the long-term benefits. If multiple actions have the same cost, the agent will select an order randomly. If a next action is found, it is executed, else a new ambition will be selected, with the current ambition taken out of consideration.

5. Framework

We have implemented the crowd and agent models described in the previous section, along with an interactive editor for both. This section discusses the resulting framework. First, an overview of the system will be given, along with descriptions of every component. Then, the editors for the two models are introduced.

5.1. System Overview

A schematic overview of our system can be seen in Figure 3. The dashed boxes indicate components that can be replaced by custom versions relatively easily. The rest of this section will give a more detailed description of the various components.

5.1.1. Simulator

The simulator is the central part of the framework, and connects the various components. It will not do anything significant itself; it only propagates the data between the different components, and allows the components to interact.

It does however contain the main update-loop of the simulation. The first step is letting the world know to update all spawn triggers, to see if any more agents should be added to the environment. After that the semantics are updated to make the environment state up-to-date. The agents themselves are updated afterwards, and finally the movement engine is requested to update the position of the agents.

5.1.2. Agent Engine(s)

The agent engines take care of the updating process of the agents, and is thus specific for the used agent model. For our agent model, the engine does not need to do much more than delegate the update process to each of the agents. However, if the agent model requires some form of communication between agents that can not be handled using semantics, the accompanying agent engine should handle that.

Detecting actions that insert ambitions into agents is the only extra inclusion in our agent engine. This detection could also be part of the agents themselves, but that would mean that the agents already know what ambitions they may eventually get.

5.1.3. Agent

The abstract agent, as defined in our simulator, requires three properties to be specified by the model-specific implementations. These properties are all related to the movement of the agents; visual information of the agents is assumed to be specified in the semantic representation that the agent controls, and any other information is specific for each agent model:

Motion Type Each agent must indicate if it takes care of the movement by itself, or if the movement engine should handle the movement. This value need not stay the same throughout the life span of the agent.

Target Position To let the movement engine know where to send the agent, each agent specifies its target position. This will not be used internally if the motion type indicates that the movement engine should leave this agent alone.

Preferred Movement Speed This property is by default the value obtained from the crowd profile. However, if the agent model decides that the agent is in a hurry, or very tired, adjusting this value will let the movement engine know to adjust the movement speed (when possible).

5.1.4. Movement Engine

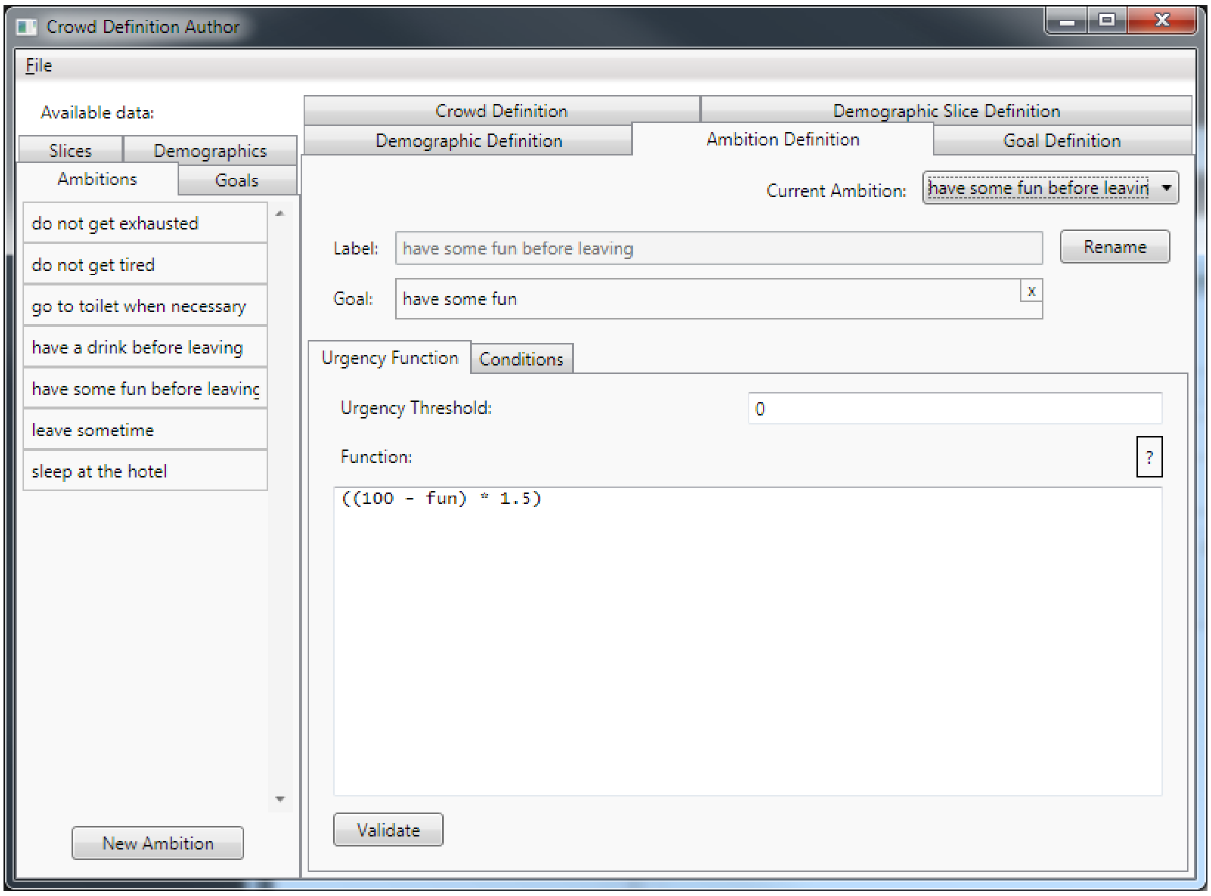
The movement engine handles the locomotion of the agents, based on their orientation and target position. Any collision avoidance should be handled here, although collision detection should be left to physics handlers provided to the semantics.

We chose to allow only a single explicit motion engine, to encourage simulating all movement with a single approach. Having multiple movement engines makes it much harder to create a realistic simulation. If a different engine is still required for part of the agents, it can still be achieved by incorporating movement into the agent engine.

The movement engine used in our system is a small wrapper around a path planner using Explicit Corridor Mapsh [31].

5.1.5. Visualizer

The visualizer is not strictly a component in the simulation system, since even without it the simulation will work. The simulator gives access to the semantic environment as well as the agents controlling the entities in that environment, which is all that is required to visualize what is happening in the simulation.



do not get exhausted
do not get tired
go to toilet when necessary
have a drink before leaving
have some fun before leaving
leave sometime
sleep at the hotel

Current Ambition: have some fun before leavin

Label: have some fun before leaving

Goal: have some fun

Urgency Threshold: 0

Function:

```
((100 - fun) * 1.5)
```

Fig. 4: Screenshot of the Crowd Profile Editor.

5.2. *Crowd Editor*

To aid in the creation of crowds and their addition to semantic virtual worlds, we have created interactive editors for the Crowd Profile and Crowd Sockets. To emphasize the independence of the two components of our crowd model, they are separate editors. The editors are also specifically built to create crowds of agents using our semantic agent model.

The Crowd Profile editor allows the user to create a crowd profile from the bottom up, but also allows users to load an existing crowd and edit it to their liking. A screenshot of the editor may be found in Figure 4, which displays the editing tab for ambitions. The left column of the editor provides aggregated lists of goals, ambitions, demographics and demographic slices, and provides easy access to them at any point in the editing process for either selection or drag and drop functionality. The rest of the editor displays what the user is currently editing.

Users can edit goals by adding, removing and editing their desires. These desires are defined by setting adesired state for entities or attributes, in addition to some explicit desires like 'leave the environment'. To set the goal of an ambition, the user only needs to drag and drop the appropriate goal from the list on the left side of the editor. The urgency valuation function of an ambition consists of a mathematical expression and a set or conditions determining if the ambition should be considered. The conditions of the ambitions are similarly defined as the desires of goals.

Demographic definitions are created by adding ambitions and specifying a probability (see also the screenshot in Figure 4). each demographic may also override the agent attribute distributions. These demographics are put into slices together with a coverage percentage of each demographic. The slices can then be added to crowds. For a single demographic, an extra spot has been reserved at the bottom. This demographic is the 'default' one, and need only be set if none of the slices cover the entire crowd.

For the Crowd Socket Editor, there are two main content collections that need to be filled: *crowd sources* and *ambition inserters*. A crowd source is a combination of a crowd profile, a spawn space and spawn trigger. The definition of conditions works similar to all other conditions used in the Crowd Profile editor. However in this case there are two sets of conditions. The first set will prevent the spawn trigger from updating, the second set will prevent it from notifying the spawn space that an agent should be spawned.

The editing tab for ambition inserters is similar to the format of the crowd profile editor. On the left side are lists with ambitions and goals, which are automatically populated with those found in any crowd used in the crowd sources. On the right side are tabs for those ambitions and goals, as well as a tab with all inserters. Each ambition inserter requires, aside from an ambition and insertion strength, a reference to a type of action and the entity type of the actor. This pair represents the actual inserter; whenever an entity of the given actor type performs an action that is a child of the given action, the given ambition will be attempted to be inserted.

6. Results

To demonstrate the viability and merits of the proposed approach, two scenarios have been created and populated by the same crowd. Section 6.1 describes the setup of the basic environment, which the two cases discussed in Sections 6.2 and 6.3 expand upon. Section 6.1 also introduces the crowd template used in both scenarios.

6.1. Basic Setup

In this section, we introduce the basic environment, the crowd to be used throughout the section, including its environment specific settings, and the initial results of populating the former with the latter.

6.1.1. The Environment

Layout & Semantics. The environment represents a part of an airport before customs. It is a hallway with benches on one side, on which agents can rest, and several establishments on the other side, including a bank (A), hotel (B), cafe (C), a concealed room with plum trees (D), and an arcade hall (E). An ATM is also present in the hallway, in the facade of the hotel. This floor plan can be seen in Figure 5a.

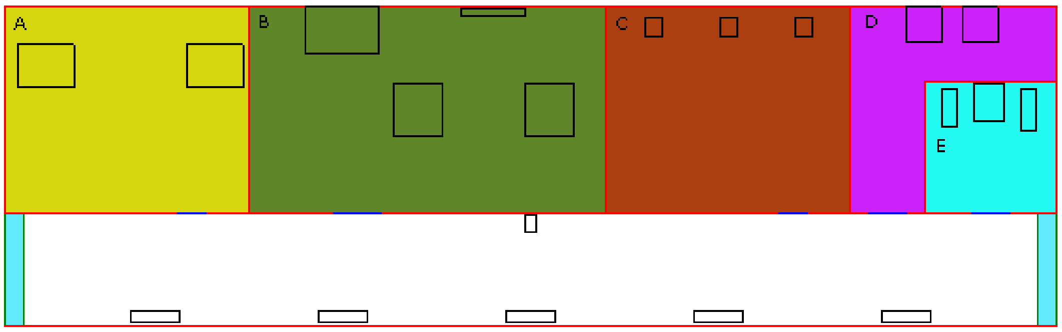
The agents enter from the left and right sides of the environment, and can exit there as well. They can also 'exit' into the hotel, however, in order to do so, they have to check in first. Checking in can be done at the front desk of the hotel, but requires a credit voucher from the bank. In order to use the lounge chairs in the hotel lobby, an agent must be checked in as well. To use the services of the arcade hall and the coffee shop, money is required, which can be retrieved at the ATM or in the bank. The plum form the trees are free. The arcades increase the enjoyment of an agent, drinking coffee sates thirst and slightly increases the need to go to the toilet, and eating plums greatly increases that need. The cafe and the bank are only usable by adults and elderly.

Crowd Socket. There is only a minimal configuration required for this environment. The spawn spaces at the ends of the hallway are specified, along with their spawn rate. Furthermore, the specific semantic entity that represents the agent is defined. Finally, the maximum number of agents in the environment is set to 50, because with any more agents the doorways would become clogged very fast.

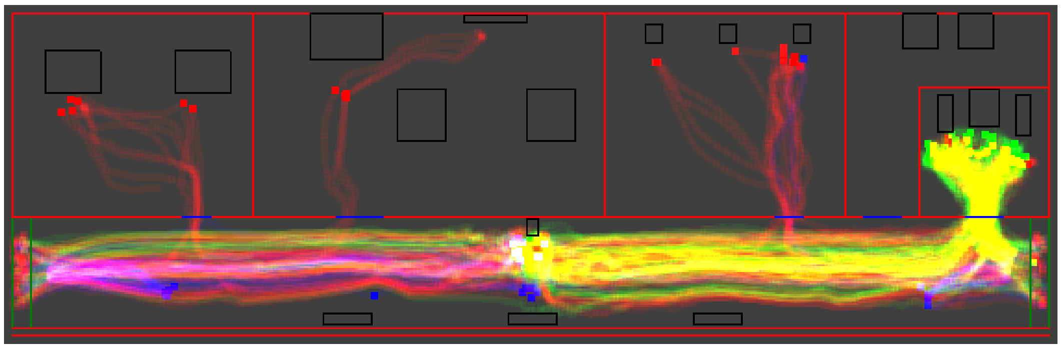
6.1.2. The Crowd Profile

Throughout the scenarios in this section we use exactly the same crowd profile. In doing so, we wish to make our point that virtual environments enriched with different semantics are able to induce rather different behaviors on the same crowd. In other words, you don't need to 're-program' the agents to have them act or react in different environments or manners.

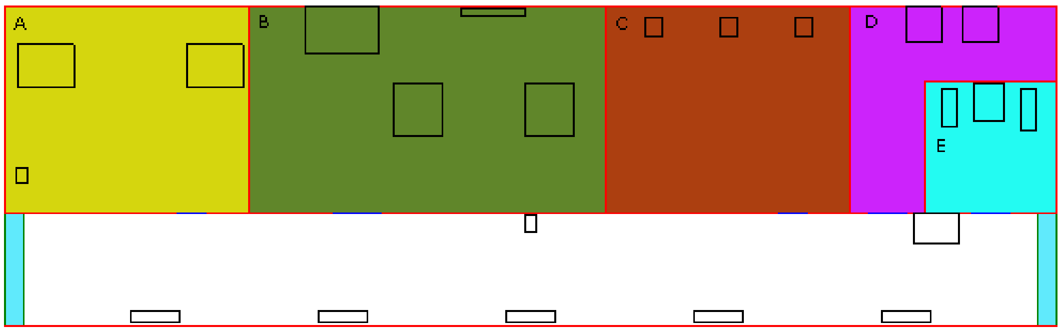
The distribution of the agents' genders is set to be equal, but in the age group distribution, adults are more likely than elderly and children: approximately 15% will be children, 15% will be elderly, and the other 70% will be adults. The average walking speed declines as the age is higher, but their determination and distractibility are equally distributed.



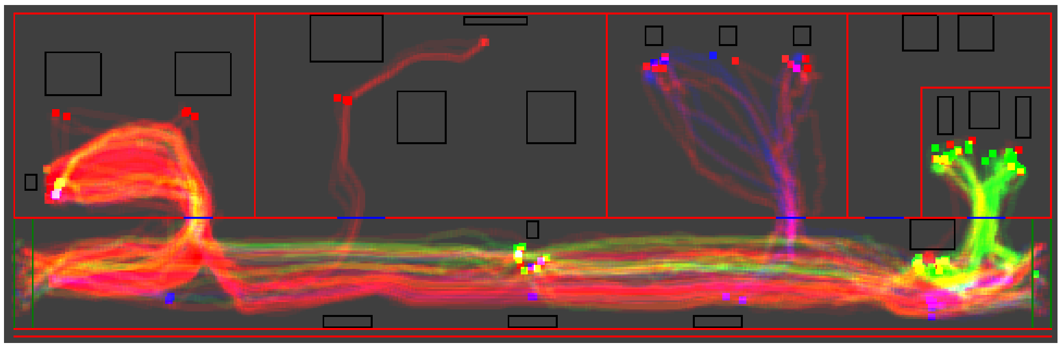
(a)



(b)



(c)



(d)

Fig. 5: On the top, the floor plan (a) and heatmap (b) of the basic setup are shown. Below, the floor plan (c) and heatmap (d) for Case 1. In both heatmaps, different colors are used for different age groups: blue traces are elderly, red traces are adults, and green traces are kids.

Every agent has a small ambition to leave the environment, such that they will never be standing still, as well as the ambition to go to the toilet when they need to. The latter increases an agent's urgency when it 'fills up'. Most children (80%) will want to do something fun before they leave, while only 10% of the adults have that ambition, and none of the elderly. All adults and elderly have the ambition to rest when the are tired, but elderly tire much faster than adults. Half of the adults and three-quarters of the elderly enter the environment in search of a drink, and half of the adults who do not want a drink aim to check in at the hotel.

6.1.3. Analysis

A heatmap of the simulation of this basic setup can be seen in Figure 5b. The easiest observation to make from this map is that the agents make no use of the available plums whatsoever. This is exactly what is expected though, as the effect of the plums does not positively influence any of the ambitions that the agents have. The agents will also never exhibit the ambition to relieve themselves, as there is nothing in the environment that they can use to do that.

Another general observation is that the agents will approximately equally distribute themselves over the various objects giving the same service. As discussed in Section 4.2, the choice of actions that agents choose to perform is based on the expected 'cost' when performing those actions. Because the calculation of the expected cost is rather crude, it is not necessarily the object closest to the entrance that gets used the most.

The heatmap also shows several agents that at some point sat on the benches. Because the average life time of an agent in the environment is low, only elderly agents have rested on the benches, adults left before they got tired. The lounge chairs are unused in this sample. While any agent can choose the lounge chairs as a possible resting place, the cost for using them is much lower when the agent is already in the check-in progress. Because the benches are spread out over the environment, agents will find it much easier to just sit on a bench, than going through the process of checking in.

Our agents take into account only one ambition at a time. Thus agents will not combine the ambition to check in to the hotel with the ambition to rest; whatever is deemed more urgent will be achieved first. Something similar could be observed with regard to the coffee shop and arcade hall. If an agent wants to visit and make use of both establishments, the agent will need to use the ATM twice (or go to the bank once) to get enough money. However, agents will never get more money if they believe they have enough to start satisfying their current ambition. Thus an agent may visit the ATM before going to the coffee shop, after which it needs to go to the ATM again before going to the arcade hall.

As long as one does not try to follow the individual agents, the system creates a fairly realistic 'background crowd'. Of course, if more detailed behavior is desired, the semantics and/or crowd definition will need to be extended accordingly.

6.2. Case 1: Hurdles

For this first case, we slightly modified the basic environment described above, in order to highlight the strength of the semantics in the world and in its objects. The same crowd is used, and the modifications to the environment were so minor that the crowd socket can also remain the same. We first describe the changes made for this new environment, after which we discuss the results of having the crowd move through the new environment.

6.2.1. The Environment

This environment is mostly the same as the environment from Section 6.1. However, in order to use the bank counters, a ticket is required from the bank's front desk in the bottom-left corner of the bank. A similar change has been applied to the arcade hall: in order to use the arcade machines, coins are needed. Money can be traded in for coins at the machine near the entrance, outside of the arcade hall. An updated floor plan can be seen in Figure 5c.

6.2.2. Analysis

Since the same crowd is used in this environment, we still see mostly the same behavior. However, as can be seen from the heatmap in Figure 5d, there are some significant differences. The major difference in the heatmap is that a disproportionate number of agents is using the front desk of the bank. Because getting a ticket is very easy, agents will (usually wrongly) assume that using the bank is faster than using the ATM. There are some agents that still use the bank after they got a ticket, but those are the same agents as the ones that checked in at the hotel a few moments later.

It is also apparent that agents often walk back and forth between the coin machine and the arcade cabinets. Agents are not able to satisfy their ambition to have 'enough' fun with just one or two plays, and their coins will usually run out before the ambition has been satisfied. Because we made our agents opt for instant satisfaction, rather than finding the optimal route to satisfy all ambitions, they will not hoard enough coins in order to keep playing until they are done. Instead, only one batch of coins is bought at a time, and a new batch is only bought when the current batch is insufficient.

Since this environment is only slightly different from the original environment, having the same crowd run in this simulation gives only a little taste of the strength of semantics for crowd simulation. By changing the way agents have to solve their problems, without changing anything to the agents themselves, we show that by employing semantics, it does not matter how convoluted the path towards the goal of the agents is. It also does not matter in how many ways the agents can fulfil their ambitions, nor what type of objects the agents need to use, nor how those objects can be used.

6.3. Case 2: Supersize

This second case uses a significantly larger environment. We also introduce ads, which will try to make agents do things that were not specified in ther crowd definition. Section 6.3.1 describes the differences between the new environment and the one used in Case 1, and Section 6.3.2 discusses the crowd behaviour in this new environment.

6.3.1. The Environment

Layout And Semantics. This environment is an extension of the environment used in Section 6.2, as can be seen in the floor map in Figure 6. The top hallway is the same as before, although there are less benches on the bottom wall to accommodate the new hallways.

The central block contains a toy store (F), toilets (G for males, J for females), juice shop (I) and a art gallery (K). The toy store will provide fun for kids while they browse, while the art gallery does the same for adults and elderly. An ad placed in the left hallway will entice agents to buy something at that toy store, although none of the agents will shop there automatically. The toilets, separated by gender, finally provide the agents with a way to relieve themselves.

Near the entrance of the toy store is a sign with information on the plum trees, and acts as an ad for those trees. In the same hallway an ad for drinks is located, which means that kids may also want a drink if they pass that ad. A fountain can also be found close by (H), although it only has an aesthetic function.

The bottom hallway contains a burger restaurant (L) and another hotel (M). An extra ATM is placed in front of the new hotel, and a generic ad for food is placed in front of the burger restaurant. The same ad is also placed in the left hallway. Agents can enter and leave from the left and right ends of the top an bottom hallways, and an ad for the hotel chain is also located at the exit near the burger restaurant.

Crowd Socket. With the addition of ads, this environment needs some more configuration before agents can use all of it. For each type of ad entity, we specify what of ambition they want the agents to have. For the plum ad, the agents will want to eat at least one plum before leaving the environment. The food ads will make agents want to eat something before they leave; in this case that can be either a plum or a burger at the restaurant. The toy store ad and hotel ad are fairly self-explanatory: they will make the agents want to buy something at the toy store and check-in to one of the hotels respectively.

6.3.2. Analysis

Behavioural Evaluation. Now that the environment contains ads, the crowd can do what it would never think of before, as can be seen on the heatmaps in Figure 7. Since the plum trees are not on the route for anything else, the fact that agents are walking near them gives enough evidence that the plum trees are now being used. The burger restaurant is eerily quiet, but that is because it costs so much. Plums will also sate the hunger of the agents, so they would rather eat a few plums than go to the restaurant.

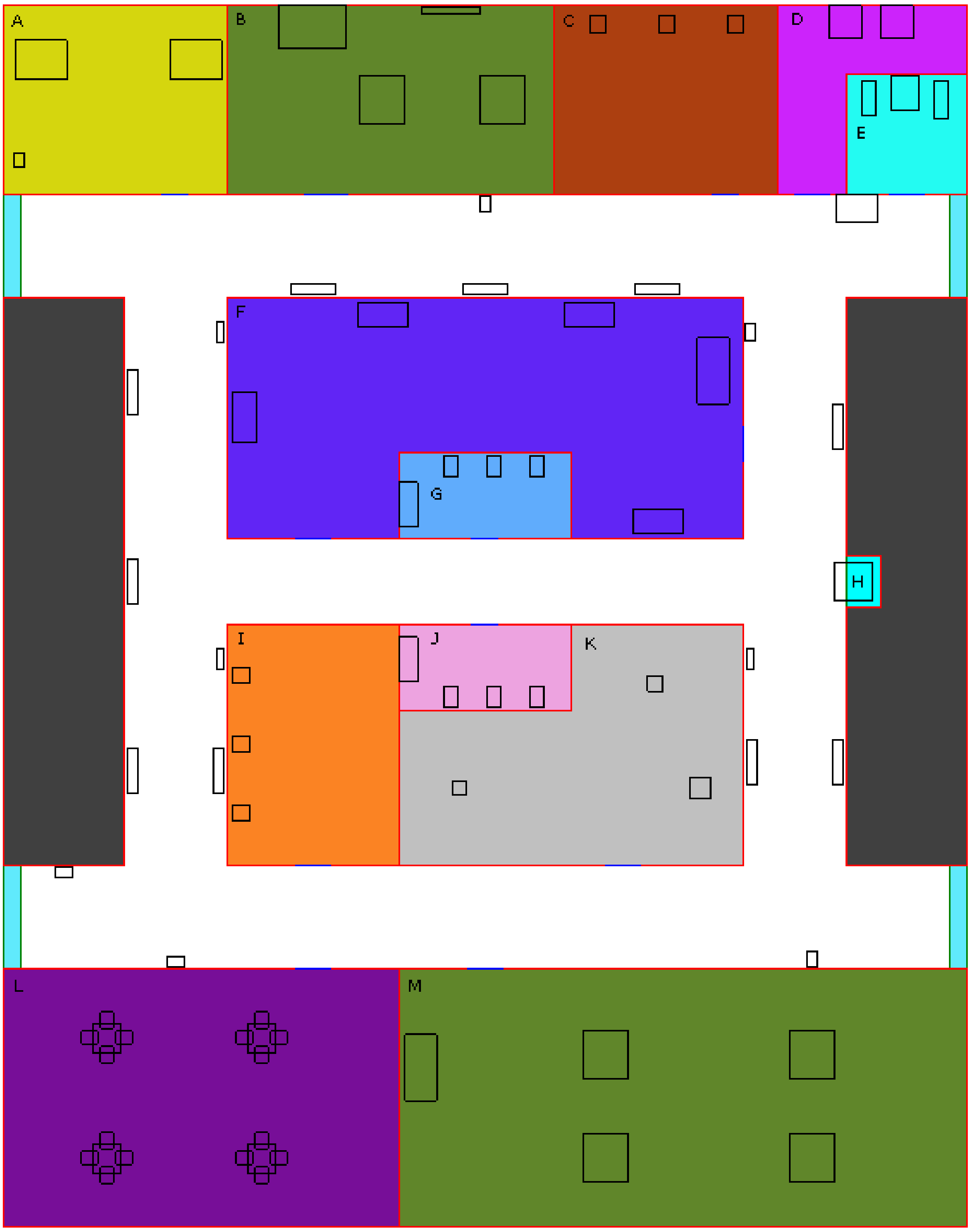


Fig. 6: Floor plan for the extended environment of Case 2.

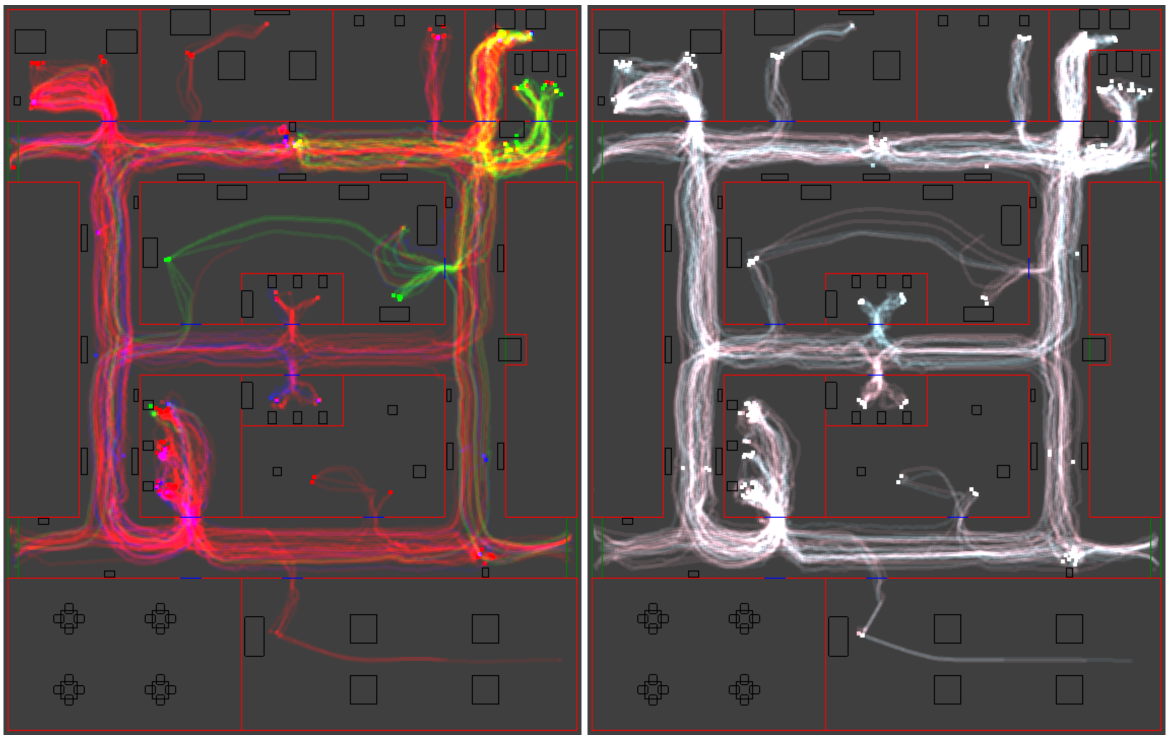


Fig. 7: Heatmaps for Case 2. The left map uses the same color scheme as the heatmaps in Figure 5; in the right map, blue traces are males and pink traces are females.

Another interesting observation is that, while the art gallery can be used by elderly, they never actually use it. This has a simple explanation, because the elderly agents will never have the ambition to do something fun before leaving. They will never have that ambition from the moment they enter the environment, because of the crowd profile, and there are no ads that will make them want to visit the gallery either.

In conclusion, now that the environment is significantly larger, the power of semantics is even more visible. Without changing anything in the crowd profile, we can let the crowd roam through this environment with just as much realism as in the smaller environments.

Performance Analysis. In the first two simulations, we could simulate approximately 175 agents without getting a noticeable lag in performance. The same amount of lag occurred in the third simulation at around 100 agents. The main bottleneck for these numbers of agents is the process of an agent finding all possible ways to satisfy its ambition. We have made very few optimizations in that algorithm however, so we expect that our framework will easily be able to handle larger crowds.

7. Conclusions

We proposed semantic crowds, a novel approach that allows one to easily define crowd templates in an easy and portable way, and re-use them without modification for virtually any environment, in which the objects available are spontaneously used in a meaningful manner. This is achieved by having each agent query the environment to find whatever objects are deemed suitable to fulfil its desires.

We briefly described our prototype system, including an interactive crowd editor that provides high-level editing parameters for defining crowd templates. We finally demonstrated how such a semantic crowd flexibly adapts its behavior to quite differently configured environments. This modular approach for crowd re-use easily combines with your custom implementations of agent behavior model and motion planner, possibly providing more fine-grained control or more realistic and detailed paths.

Acknowledgements

We thank Roland Geraerts [31] and Atlas Cook IV for kindly providing us with their path planner library and promptly assisting us with its integration in our prototype.

References

[1] T. Tutenel, R. Bidarra, R. M. Smelik, K. J. de Kraker, The role of semantics in games and simulations, Computers in Entertainment 6 (4) (2008) 1–35.

[2] J. Kessing, T. Tutenel, R. Bidarra, Services in game worlds: A semantic approach to improve object interaction, in: ICEC '09: Proceedings of the 8th International Conference on Entertainment Computing, LNCS vol. 5709, Springer-Verlag, 2009, pp. 276–281.

[3] M. Luck, R. Aylett, Applying artificial intelligence to virtual reality: Intelligent virtual environments, Applied Artificial Intelligence 14 (1) (2000) 3–32. doi:10.1080/088395100117142.

[4] N. Farenc, R. Boulic, D. Thalmann, An informed environment dedicated to the simulation of virtual humans in urban context, Proceedings of Eurographics'99 18 (3) (1999) 309–318.

[5] K. Otto, F. U. Berlin, Towards semantic virtual environments, in: Workshop Towards Semantic Virtual Environments, 2005, pp. 47–56.

[6] G. Thomas, S. Donikian, Modelling virtual cities dedicated to behavioural animation, Computer Graphics Forum 19 (3) (2000) 71–80. doi:10.1111/1467-8659.00399.
URL `http://dx.doi.org/10.1111/1467-8659.00399`

[7] M. Badawi, S. Donikian, The generic description and management of interaction between autonomous agents and objects in an informed virtual environment, Computer Animation and Virtual Worlds 18 (4-5) (2007) 559–569.

[8] M. Kallmann, D. Thalmann, Modeling objects for interaction tasks, in: Proceedings of the Eurographics Workshop on Animation and Simulation, 1998, pp. 73–86.

[9] M. Gutierrez, F. Vexo, D. Thalmann, Semantics-based representation of virtual environments, International journal of computer applications in technology 23 (2) (2005) 229–238.

[10] T. R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (1993) 199–220.

[11] F. Grimaldo, M. Lozano, F. Barber, G. Vigueras, Simulating socially intelligent agents in semantic virtual environments, The Knowledge Engineering Review 23 (04) (2008) 369–388. arXiv:http://journals.cambridge.org/article_S026988890800009X, doi:10.1017/S026988890800009X.
URL `http://dx.doi.org/10.1017/S026988890800009X`

[12] H. Jiang, W. Xu, T. Mao, C. Li, S. Xia, Z. Wang, A semantic environment model for crowd simulation in multilayered complex environment, in: Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology, VRST '09, ACM, New York, NY, USA, 2009, pp. 191–198. doi:http://doi.acm.org/10.1145/1643928.1643972.
URL `http://doi.acm.org/10.1145/1643928.1643972`

[13] A. Treuille, S. Cooper, Z. Popović, Continuum crowds, ACM Trans. Graph. 25 (2006) 1160–1168. doi:10.1145/1141911.1142008.

[14] C. W. Reynolds, Flocks, herds and schools: A distributed behavioral model, SIGGRAPH Comput. Graph. 21 (1987) 25–34. doi:10.1145/37402.37406.

[15] X. Tu, D. Terzopoulos, Artificial fishes: physics, locomotion, perception, behavior, in: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, SIGGRAPH '94, ACM, New York, NY, USA, 1994, pp. 43–50. doi:10.1145/192161.192170.

[16] S. R. Musse, D. Thalmann, Hierarchical model for real time simulation of virtual human crowds, IEEE Transactions on Visualization and Computer Graphics 7 (2001) 152–164. doi:10.1109/2945.928167.

[17] B. Ulicny, D. Thalmann, Crowd simulation for interactive virtual environments and VR training systems, Computer Animation and Simulation 2001 (2001) 163–170.

[18] N. Farenc, S. Musse, E. Schweiss, M. Kallmann, O. Aune, R. Boulic, D. Thalmann, One step towards virtual human management for urban environment simulation, in: Proceedings of the ECAI workshop on intelligent user interfaces, Vol. 3, Citeseer, 1998.

[19] T. Abaci, J. Ciger, D. Thalmann, Planning with smart objects, in: Proceedings of the 13th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision: WSCG, 2005, pp. 25–28.

[20] E. Bouvier, E. Cohen, L. Najman, From crowd simulation to airbag deployment: particle systems, a new paradigm of simulation, Journal of Electronic Imaging 6 (1) (1997) 94–107. doi:10.1117/12.261175.

[21] D. Helbing, P. Molnar, Self-organization phenomena in pedestrian crowds, Arxiv preprint cond-mat/9806152.

[22] D. Helbing, I. Farkas, T. Vicsek, Simulating dynamical features of escape panic, Nature, Vol. 407, pp. 487-490, 2000doi:10.1038/35035023.

[23] A. Braun, S. Musse, L. de Oliveira, B. Bodmann, Modeling individual behaviors in crowd simulation, in: Computer Animation and Social Agents, 2003. 16th International Conference on, IEEE, 2003, pp. 143–148. doi:10.1109/CASA.2003.1199317.

[24] N. Pelechano, J. Allbeck, N. Badler, Controlling individual agents in high-density crowd simulation, in: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, 2007, pp. 99–108.

[25] J. Kessing, T. Tutenel, R. Bidarra, Designing semantic game worlds, in: PCG 2012: Proceedings of the 3rd workshop on Procedural Content Generation for Games, 2012.

[26] T. Tutenel, R. M. Smelik, K. J. de Kraker, R. Bidarra, Using semantics to improve the design of game worlds, in: Proceedings of AIIDE '09, the 5th Conference on Artificial Intelligence and Interactive Digital Entertainment, Stanford, CA, USA, 2009.

[27] T. Tutenel, R. Bidarra, R. M. Smelik, K. J. de Kraker, A semantic scene description language for procedural layout solving problems, in: Proceedings of AIIDE '10, the 6th Conference on Artificial Intelligence and Interactive Digital Entertainment, Stanford, CA, USA, 2010.

[28] T. Tutenel, R. M. Smelik, R. Lopes, K. J. de Kraker, R. Bidarra, Generating consistent buildings: a semantic approach for integrating procedural techniques, IEEE Transactions on Computational Intelligence and AI in Games 3 (3) (2011) 274–288.

[29] D. de Paiva, R. Vieira, S. Musse, Ontology-based crowd simulation for normal life situations, in: Proceedings of Computer Graphics International Conference, IEEE Computer Society, Los Alamitos, CA, USA, 2005, pp. 221–226. doi:10.1109/CGI.2005.1500421.

[30] M. E. Bratman, Intention, Plans, and Practical Reason, Harvard University Press, 1987.

[31] R. Geraerts, Planning short paths with clearance using explicit corridors, in: Proceedings of Robotics and Automation (ICRA), 2010 IEEE International Conference on, IEEE, 2010, pp. 1997–2004.