A Procedural Approach to Simulate Virtual Agents Behaviours in Indoor Environments

Laura M. Flach, Fernando P. Marson, Vinícius J. Cassol and Soraia Raupp Musse
Laboratório de Simulação de Humanos Virtuais - VHLab
Programa de Pós-Graduação em Ciência da Computação - PPGCC
Pontifícia Universidade Católica do Rio Grande do Sul - PUCRS - Brasil
Web page: http://www.inf.pucrs.br/~vhlab/
E-mail: {laura.flach, fernando.marson, vinicius.cassol}@acad.pucrs.br, soraia.musse@pucrs.br

Abstract—**This paper presents a procedural model to automatically simulate virtual agents and their behaviors in indoor environments. The work is focused on creating groups and families of virtual agents and their actions inside their houses without user intervention. The groups are created based on semantic information from the environment to be populated and on data from statistics of real life. The model generates behaviors for each group member. Such behaviors should be coherent to the environment, to the group's and member's characteristics and also to the time. Parameters are coded into seeds, which makes possible the persistence of the data (the possibility of the regeneration of the context at any moment) and simple execution (the model is fully automatic). Results indicate that our model is suitable for background characters in applications such as games and automatic content generation.**

I. Introduction

In games such as Grand Theft Auto-GTA IV [1] some buildings and houses can not be used to interact since they were not modelled by the game designers. In huge urban environments such as the world of GTA, many are the spaces where NPCs (non-player characters) can live and evolve. However, only those which were modelled by designers can be populated, even if we are talking about simple and background characters and their actions. The difficulty is mainly due to the fact that each space or object should be modelled manually by designers, so they focus on main actions. Other examples of such type of games are Assassin's Creed [2] e World of Warcraft [3]. Another application is the recent requirements for generating automatic content at runtime that could be applied in TV, movies as well.

The problem to be dealt in this paper is concerned with the automatic generation of indoor environments, groups and families of virtual agents which should serve as background characters in games and other computer graphics applications. Moreover, such entities in a game should behave accordingly some requirements. Firstly, environments should be generated in a coherent way in order to be populated. Although some authors have proposed nice models to generate procedurally terrains, cities, ocean, buildings and other urban components [4], [5], [6], [7], [8], we were inspired in the work proposed by Marson [9] which generates floor plans in geometric and semantic format. Here the coherence is mainly concerned with the presence of specific rooms into the spaces, e.g. kitchen, toilete, bedrooms and their localization into the house (this will be detailed in Section III).

Secondly, the population inside a building (groups or families) should be coherent to the environment. For instance, a house with two bedrooms has low probability to serve a family with 6 persons, even if it can be possible. Also, once we have the group or family, the membership should also be consistent. A family of five children without an adult is not coherent, for instance. Here, we would like to emphasize that all possibilities can obviously exist in real life, depending on the country or culture. However, we adopt the rules based on common sense and statistics of some cities which are indeed public available[1].

In addition to the coherence of environment, groups and members, our model also generates the people actions as a function of time. Here an important aspect of procedural modelling [10] was considered, the entities can be generated and evolved only if the user/player is viewing that. It reduces the time of processing and the amount of memory needed to process a huge environment and also allows to generate automatic content at runtime. However, the challenge here is to be able to generate/regenerate the entities considering that time has passed but in same time preserving the past.

This paper is organized as follows: in next section we describe some few papers concerned with the modelling of behaviors in a procedural and automatic way. In Section III the details of our model are presented while Section IV shows results obtained with our work. Finally, in Section V some final considerations and ideas for future work are discussed.

II. Related work

Procedural modelling can be understood in a vaste range of applications. In this section we present some few papers that use procedural modelling in order to provide automatic behaviors to virtual agents.

The pioneer paper in procedural animation of groups is certainly the work proposed by Reynolds [11]. This describes agents endowed with abilities to navigate around their world in a life-like and improvisational manner. Combinations of steering behaviours can be used to achieve higher level goals, for example: get from here to there while avoiding obstacles, follow this corridor, join that group of characters and etc. Hidalgo et al [12] presents a model that helps to include procedural generation into existing modelling and rendering

[1]http://www.censo2010.ibge.gov.br

applications. Due to its design, extensibility and comprehensive interface, their technique can handle user objects to create and improve applications with procedural generation of content.

Several aspects of group behaviors have been treated in last years. A specific problem related to path planning is the generation of realistic motion along the path. Lavalle [13] introduced the concept of a Rapidly-exploring Random Tree (RRT) as a randomized data structure for path planning problems. An RRT is iteratively expanded by applying control inputs that drive the system slightly toward randomly-selected points. Choi et al. [14] proposed a model based on a probabilistic path planning and hierarchical displacement mapping to generate a sequence of realistic movements of a human-like biped figure to move from a given start position to a goal with a set of prescribed motion clips. Metoyer and Hodgins [15] proposed a method for generating reactive path following based on the user's examples of the desired behavior.

Rodriguez et al. [16] proposed a heuristic approach to planning in an environment with moving obstacles using dynamic global roadmap and kinodynamic local planning. Kallmann and Mataric [17] proposed dynamic roadmaps for online motion planning in changing environments. When changes are detected in the workspace, the validity state of affected edges and nodes of a precomputed roadmap are updated accordingly.

More specifically concerning the motion of groups, Kamphuis and Overmars [18] introduced a two-phase approach, where a path for a single agent (a backbone path) is generated by any motion planner. Next, a corridor is defined around the backbone path and all agents will stay in this corridor. Lien and collaborators [19] proposed ways for using roadmaps to simulate a type of flocking behavior called shepherding behavior in which outside agents guide or control members of a flock. Recent work were developed aiming to produce coherently and realistically group behaviors taking into account steering and formation of groups. The way the group members interact with others individuals and groups is presented in [20]. In such work, the velocity space to plan the avoidance maneuvers of each group is used to maintain a configuration that facilitates the social interactions between the group members.

In comparison with our approach, we are focused on automatic generation of groups and families of virtual agents in a plausible semantic way. In addition, their actions and behaviors are generated based on the group/family they are part and the environment they are evolving on. An important contribution from our model is the persistence over the time which uses seeds codification optimizing memory usage and time processing. Moreover, as far as we know this is the first completly fully automatic method to generate families of agents and their behaviors in indoor environments, considering persistence and coherence of data generation. Next section details our model.

III. The Procedural Model

The proposed model uses the connection between environment and population as a key element in the definition of agents and their behaviors. It uses seeds to keep the persistence of information, i.e. based on such seeds it is possible to generate again the same family and members to a specific house. It also uses the time in order to keep the coherence of behaviors. These information is coded on seeds that maintain the needed data to provide these two important characteristics in procedural models: persistence and coherence. The overview of our model is illustrated in Figure 1. In this figure it is possible to see that three seeds are used as the input for creation of families/groups, agents and their behaviours. Also, rules defined based on statistics of real cities have been used as well. In addition, the time element is used in order to determine the behaviours. Next sections describe details of the hole process.

A. The Environment

Firstly, we use the model proposed by Marson [9] to create a house. Some parameters are needed as height, length and width of the building. It is also necessary to inform the list of desired dimensions and functions for each room. One example of generated floor plan can be viewed in Figure 2. Each floor plan also requires a position in the 3D space where its center should be located. It works to place different floor plans in a virtual city.

In addition to geometrical information, the process of environmental generation creates a seed. It is useful to provide the persistence (be able to generate again the same environment). In our case, the fields of environmental seed are: number of rooms and bedrooms, total area, minimum and maximum coordinates which represent the bounding box of the building. This seed (Figure 3) is input for family and agents creation, as discussed in next sections.

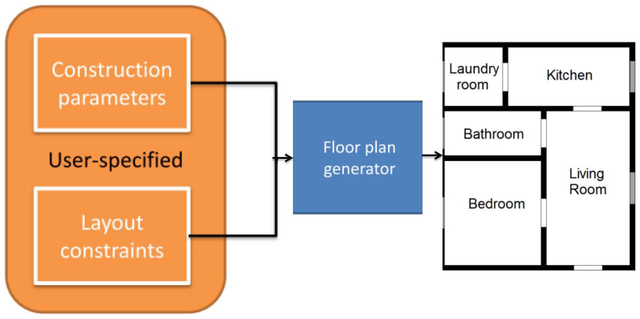
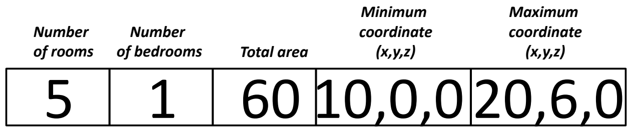


Fig. 2. The generation process of floor plans. Construction parameters and layout constraints are provided by the user as input.



Number of rooms	Number of bedrooms	Total area	Minimum coordinate (x,y,z)	Maximum coordinate (x,y,z)
5	1	60	10,0,0	20,6,0

Fig. 3. Environmental seed generated by the floor plan generator.

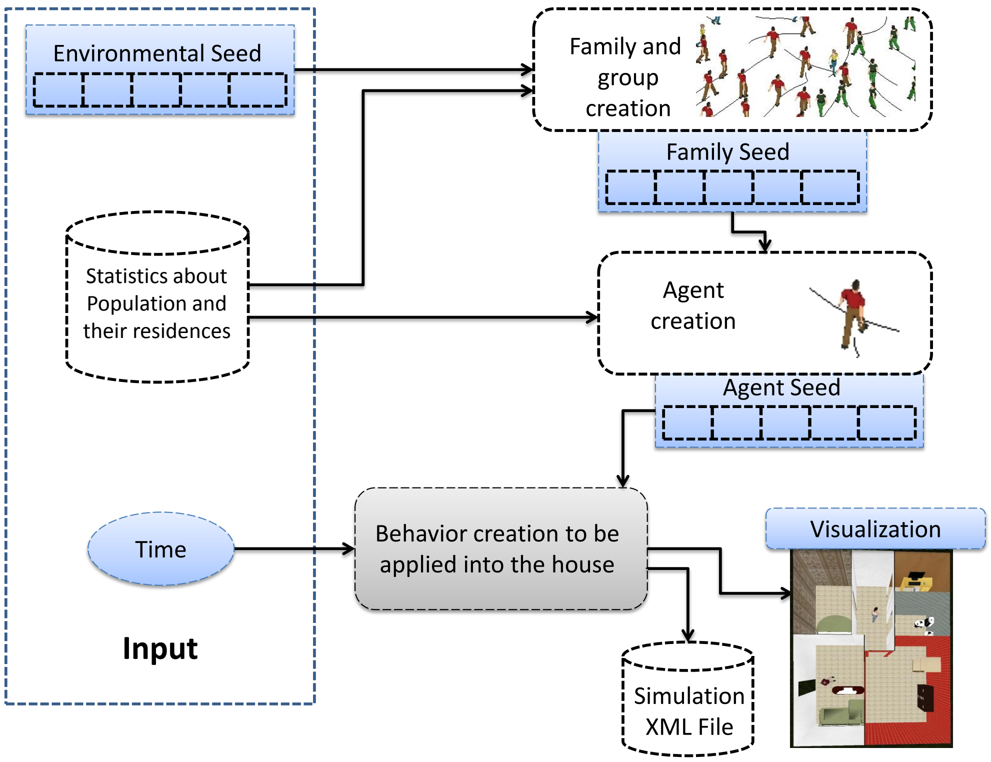

Environmental Seed
Family and group creation
Family Seed
Statistics about Population and their residences
Agent creation
Agent Seed
Time
Behavior creation to be applied into the house
Visualization
Simulation XML File
Input

Fig. 1. Overview of our model.

Number of Family Members	Specified percentages for Random Process
1	22.65
2	30.18
3	24.45
4	16,07
5	6.65

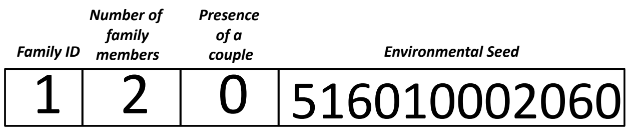
TABLE I. PERCENTAGES OF NUMBER OF MEMBERS EXISTENT IN FAMILIES ACCORDING TO FOUND IN STATISTICAL INFORMATION.

B. Family Creation

The family is a group of agents which lives in the environment defined in last section. So, we use the environmental seed, as illustrated in Figure 1, to control the randomness of generation using acceptable rules as parameters. Such rules have been specified using statistics available about real cities [2] which defines the percentages of families size existent until 5 members, as illustrated in Table I [3] and the presence of a couple (we used %36,57 to have a presence of couple of any gender).

To create a family we use also the number of bedrooms defined in the environmental seed. Based on statistics, we mapped simple possibilities of having a certain number of persons in the family when their house has a specific number of bedrooms. Then, the random process happens generating a cerain family containing i members.

Although we do not find any official information about number of people in families correlated with number of bedrooms in the houses, we defined empirically two simple rules in order to associate agents with their bedrooms [4]: $i)$ If there is only one bedroom, all members sleep at such room and $ii)$ If there are two or more bedrooms and there is a couple, the couple is associated to one bedroom and the others are shared by the rest of family uniformly distributed. Each created family receives an ID which is unique. Then, the family seed is created which fields are: ID of family, number of members, the presence of a couple (0/1) and the environmental seed. This last component is included because each family contains also information about the house they live. Figure 4 illustrates the family seed. Next section describes the generation of members for a specific created family.



Family ID	Number of family members	Presence of a couple	Environmental Seed
1	2	0	516010002060

Fig. 4. Family seed considering $ID = 1$, 2 members and without a couple.

C. Agent Creation

Even if any family can be accepted, we believe some configurations are not accepted, e.g. a family containing only kids.

[2]http://www.censo2010.ibge.gov.br.

[3]We decided to generate families of maximum five members due to low probability to have other sizes.

[4]It is important to keep coherence in the actions, e.g. agents should go to sleep in the correct bedroom.

The family seed is used to generate characteristics of family members. Let be age, gender, role in the family and schedule (preferred period of the day to be at home) as a vector of fixed attributes ($\vec{a}$) of individuals which possible values are specified as follows:

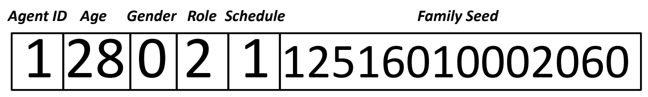
- $\vec{a}_{age} = \{0, 1\}$ (children or adult)
- $\vec{a}_{gender} = \{0, 1\}$ (female or male)
- $\vec{a}_{role} = \{0, 1, 2\}$ (responsible, student or relative)
- $\vec{a}_{schedule} = \{0, 1\}$ (night or morning)

The family seed controls the randomness of possibilities to define ($\vec{a}$) applied in the statistics data. In this case, the database contains information about who should be the members in a certain family composed by a certain number of members, as illustrated in Table II. Similar tables are used for age, role and schedule.

Number of Members	Percentages for Male	Percentages for Female
1	39.39%	60.61%
2	48.00%	52.00%
3	55.12%	44.88%
4	59.75%	40.25%
5	55.91%	44.09%

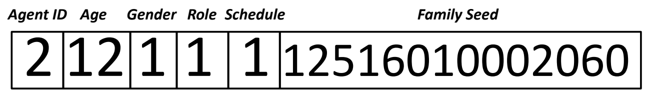
TABLE II. PERCENTAGES OF GENDER OF MEMBERS ACCORDING TO STATISTICAL INFORMATION.

For instance, in Section III-B our example family, as defined in family seed, is composed by two members. After the random process initialization we define $\vec{a}_i$ for all i agents. A simple rule is applied determining that all families should have at least one adult and it impacts the age of at least one member, and his/her role which must be responsible. In addition, if the family contains a couple, both agents should be responsible and adults. For other attributes we use exclusively the statistic information. $\vec{a}$ impacts the visualization as well as the action selection and its occurrence in the time, as discussed in next section. Moreover, an agent seed is determined using $\vec{a}$ and the family seed, as illustrated in Figure 5. Agent seed is used in next steps. Another member of family example (Figure 4) was created and represented in Figure 6.



Agent ID	Age	Gender	Role	Schedule	Family Seed
1	28	0	2	1	12516010002060

Fig. 5. Agent seed considering $ID = 1$, an adult female, role=relative and main schedule=diurnal.



Agent ID	Age	Gender	Role	Schedule	Family Seed
2	12	1	1	1	12516010002060

Fig. 6. Agent seed considering $ID = 2$, a male aging 12 years old, role=student and main schedule=diurnal.

In addition to $\vec{a}$ (which represent fixed attributes), each agent i has a set of variable status $\vec{s}$ which change as a function of time. They are: Hunger(H), Energy(E), Mood(M), Hygiene(Hy) and Physiological needs(P). Theses status evolve as a function of time based on certain curves (standard behaviors of variable status represented by $\vec{f}$) that are illustrated in Figures 7, 8, 9, 10 and 11 for status hunger, energy, mood, hygiene and physiological needs respectively. These graphics have two curves depending on the adopted schedule for each agent and were empirically defined. Indeed, $\vec{f}$ define the standard pattern of agents status in order to initialize them and evolve in the simulation scenario. In Section III-D we show how the action selection uses such information.

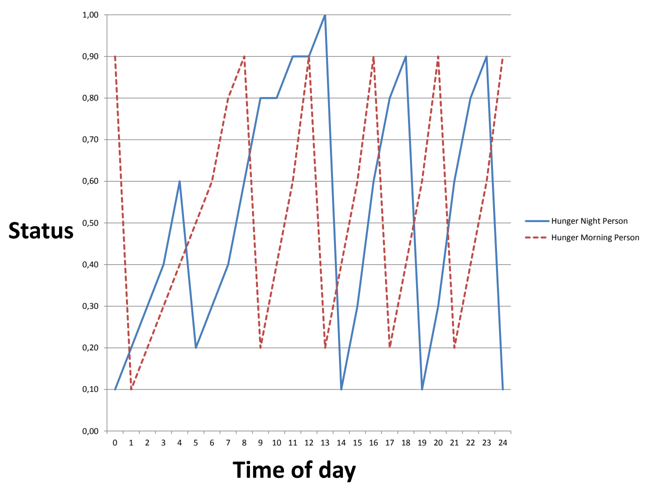


Fig. 7. Standard Behaviour for status Hunger as a function of period of the day.

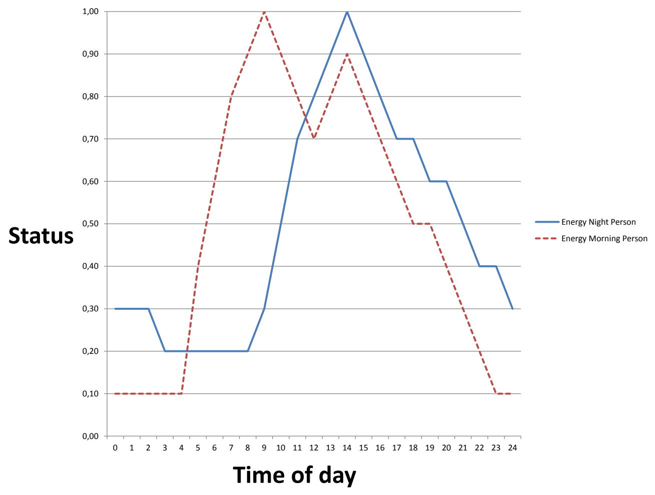


Fig. 8. Standard Behavior for status Energy as a function of period of the day.

D. Action Selection

Once environment, family, members and their attributes and status are created, we are able to select their behaviors. The fixed attributes $\vec{a}$ of family members as specified in

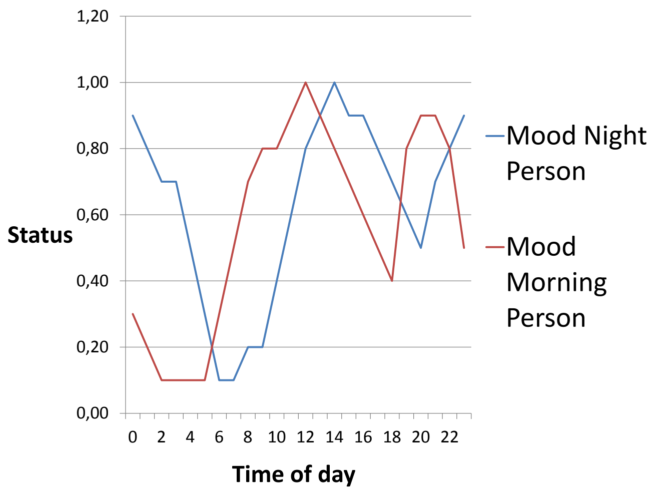


Fig. 9. Standard Behaviour for status Mood as a function of period of the day.

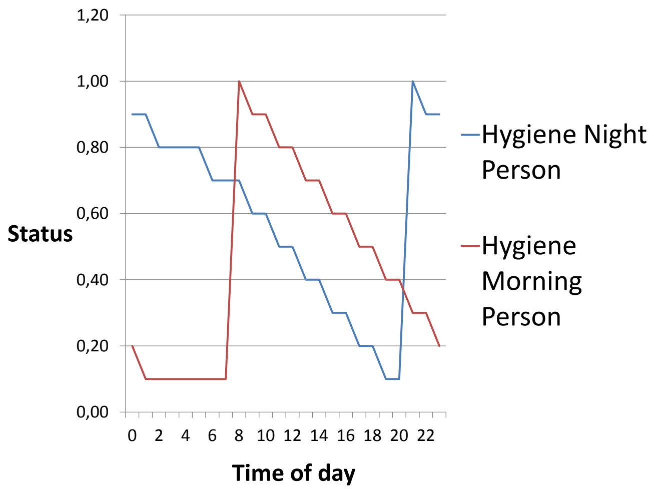


Fig. 10. Standard Behaviour for status Hygiene as a function of period of the day.

Section III-C are used to basically two functions. Firstly, age and gender are used to determine the visualization aspects [5]. While role and schedule are used to define which agents should be at home during different periods of the day, e.g. students and responsibles can be at their school and work in the determined schedule. We determined that schedule morning represents that people can be outside home from 8:00 to 12:00/from 8:00 to 18:00/from 14:00 to 18:00. In the same way the schedule night means be outside home from 14:00 to 22:00/from 18:00 to 24:00.

The variable status $\vec{s}$ defines the action the agents can adopt once they are at home. In this case we use deterministic finite state machine (FSM) [21] defined by a quintuple $(\sum, J, j_0, \delta, F)$, where:

- $\sum$ is the input alphabet (K, B, L, Ba). We defined the alphabet as the first letter of the room in the house

[5]we could also use to determine behaviours, but we did not find statistics information to validate them

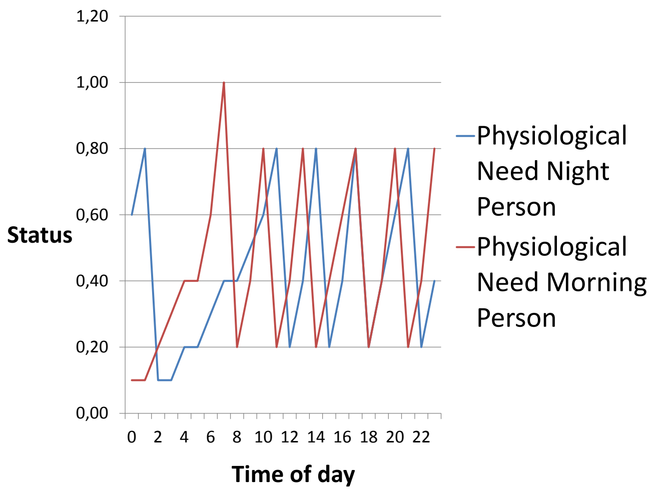


Fig. 11. Standard Behaviour for status Physiological Needs as a function of period of the day.

where actions can occur, depending on the status $\vec{s}$ (Kitchen, Bedroom, Living room and Bathroom).

- J is a finite, non-empty set of states.
- j_0 is an initial state, an element of J. In this case, agents should start at any place at the house.
- δ is the state-transition function, i.e. $\delta : JX\sum \rightarrow J$. This function depends on levels of attributes (H, E and etc).
- F is the set of final states, indeed equal to J.

Once one agent should be instantiated by first time $t = 0$, $\vec{s_t}$ is generated based on their standard behaviors $\vec{f}$ and a random process using the agent seed (Figure 5), as Equation 1:

$$\vec{s_t} = Random(agentseed, \vec{f_t}). \tag{1}$$

where $\vec{s_t}$ is related with agents attributes (hunger, energy and etc) and $\vec{f_t}$ is concerned with standard behavior of attributes. Then, simulation can start by firstly placing agents at home and initializing the FSM with first state J, i.e. one of following places: (K, B, L, Ba). A simple rule determines that J of agent i can not be Ba if another agent k is currently at bathroom $(J_k = Ba)$. When agents are in the determined places, their attributes change according to Table III. As a function of time, two functions are applied, firstly the variation of attributes in next time $(t+1)$ is performed based on standard behaviors based on the equation 2:

$$\vec{s}_{t+1} = \vec{f}_{t+1} + (\vec{f_t} - \vec{s_t}). \tag{2}$$

Secondly, the transition function δ is applied by detecting levels of status that are lower than a threshold (we empirically defined 20% of high value [6]). When it happens, the states are changed. For instance, if $Hy_t < \frac{Hy_{high}}{5}$ then $J = Ba$. Obviously, conflicts can happen, e.g. when two attributes are

[6]s_t adopts values between 0.0 and 1.0

simultaneously lower than minimum threshold. To solve such problems we determine a list of action priorities that are suitable to solve conflicts as follows: (P, Hy, H, E, M). On the other hand, if an action finishes, but any attribute is lower than threshold in order to activate a new action, the agents chooses randomly (using agent seed) the next status.

Since our simulation is presented visually in the viewer, actions should take a time to be accomplished. In this case, we consider default values for duration time (which are also randomly specified for each agent based on its seed) in order to be executed (see Table III). During the simulation, only actions executed in living room and bedroom can be interrupted by others with higher priority.

Status $\vec{s}$	Status J in FSM	Duration Time (mins)	New value for $\vec{s}$
P	Ba	20	0
Hy	Ba	40	1
H	K	50	0
E	B	300	1
M	L or B	120	1

TABLE III. DURATION OF ACTIONS EXECUTED IN SIMULATION AND VALUES ADOPTED SINCE AGENTS REACH A DIFFERENT STATUS J IN THE FSM.

The time is an important aspect in this type of simulation since user can decide to go forward, e.g. when agents are sleeping. In this case, user can decide change the time level of detail to one of 3 possible configurations: 1 frame can be $= \frac{1second}{30}$, $= \frac{1minute}{30}$ or $= \frac{1hour}{30}$.

Also, since this model is suitable for background characters, user can be interacting in another place in the virtual city and not seeing what is going on into a specific house. Consequently, we turn off the visualization and the processing of agents into the houses that are not being viewed by the user. However, the user can any time to go back again to see any house and agents. The challenge here is to re-instantiate the simulation entities in a coherent way as well as considering the past time. Firstly, all seeds are saved and in addition we save the current status $\vec{s}_{st}$ of each agent in the very last frame it was visualized as well the time of such recording (here indicated at saved time st). If the house should be re-instantiated at a specific time t, we firstly generate the environment, then the family using environmental seed (same family is generated) and fixed status of agents ($\vec{a}$) (using family seed) as well. To specify the variable agents status $\vec{s}_t$ for each agent at time t, we consider the last saved status of each agent together with standard curves $\vec{f}$, according to Equation 2, but replacing t by st. Figure 15 illustrates the process of re-instantiation (at 10:00) of energy attribute of a specific agent when compared to the standard behaviour of energy.

Table IV shows variable status for members in the example family already illustrated in Figures 4, 5, 6 when firstly instantiated at 14:00.

Attributes	Agent ID=1	Agent ID=2
Physiological Needs	0.3	0.4
Hygiene	0.4	0.45
Hunger	0.1	0.2
Energy	0.8	0.9
Mood	0.9	0.9

TABLE IV. INSTANTIATED ATTRIBUTES FOR MEMBERS OF FAMILY EXAMPLE AT 14:00.

E. *Agents Animation*

Since we know which action should be performed, we need to compute the agents motion. It considers the agent's current positions and a position of a specific place at the house, where the action can be executed. Both positions are used as input for a path planning as proposed by Cassol et al. [22]. With main goal to simulate virtual agents into a house environment, we work with two main modules: path planning and agents motion. To provide path planning, all the rooms of the house as well the doors (responsible to connect the rooms) are considered as nodes in a connective graph. Such graph, illustrated in Figure 12, is used as input for the A* Path Planning Algorithm. For instance, if the agent is currently in the living room and need to perform an action in the kitchen, the result of path planning will indicate that the agent need to cross the kitchen's door and then enter the kitchen.

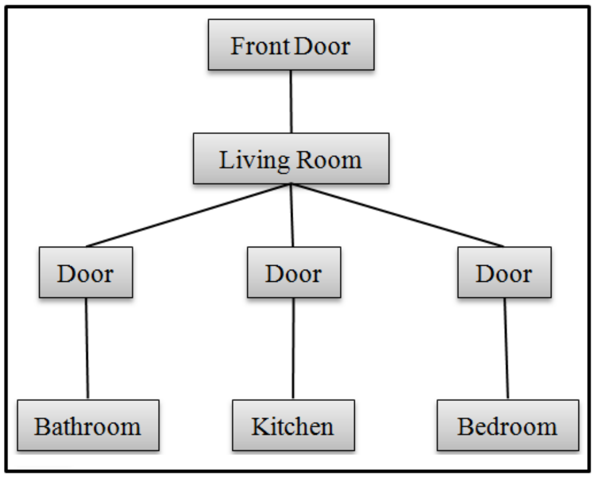


Fig. 12. Connective graph used as input for the path planning algorithm.

After have defined the best path the agent should follow to reach its desired room, we are able to compute the agents motion. To perform this, we represent the spaces in a virtual world by a set of points called markers (dots in the space) [23] which are also used to avoid collisions among agents and obstacles. However, there are different types of markers) [22] that should impact how the virtual characters move in the environment. In this work, we use non-walkable markers which goal is to define the regions of the environment where agents should not walk (i.e. obstacles).

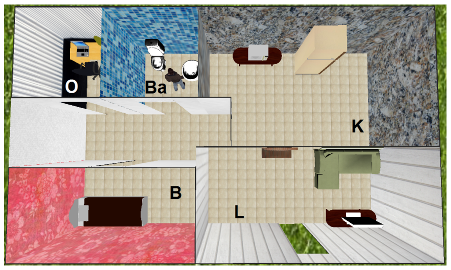
IV. EXPERIMENTAL RESULTS

This section describes some fully automatically generated environments and families using our model. We generated three different houses represented by three seeds sumaryzed in Table V. Illustrations of the houses are showed in Figure 13.

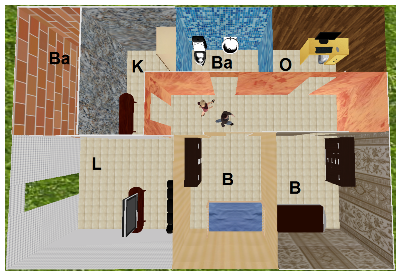
Rooms	Bedrooms	Area	Min. Coord.	Max. Coord.
5	1	43	-15.0, 5.0, 0.0	-9.0, 12.0, 0.0
7	2	63	-3.0, 5.0, 0.0	4.0 , 14.0, 0.0
8	3	67	10.0, 5.0, 0.0	17.0, 14.0, 0.0

TABLE V. ENVIRONMENTAL SEEDS FOR THREE GENERATED HOUSES.

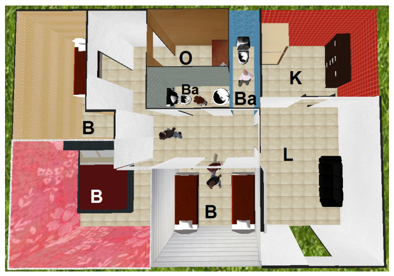
We used two copies of each house placing them in different positions in the virtual space and consequently having



(a)



(b)



(c)

Fig. 13. Environments generated with our model to be populated with virtual agents: Houses countaining 1(a), 2(b) or 3(c) bedrooms. Capital letters indicate the rooms.

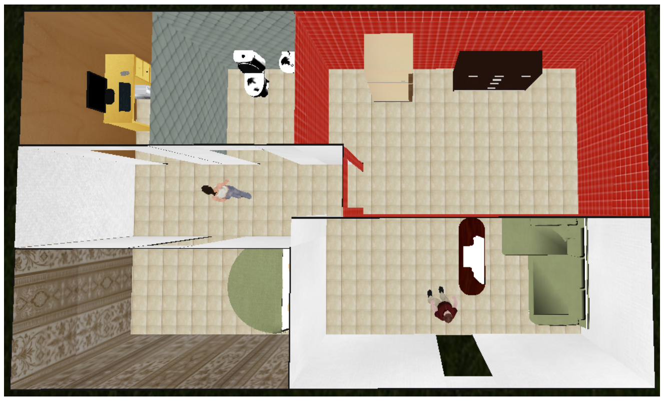
six different family seeds. Each one was used to generate automatically a family. In the Table VI we detailed two families from six generated ones. Second and third column are concerned with family ID=1 that lives in the house with only one bedroom. This family is formed by a couple, and their variable status are described in the table once instantiated at 20:00. In the other columns we present the attributes of another family generated to live in the house with 3 bedrooms. This family is formed by a couple (female and male), a relative male and a child male and the attributes indicate the values when family was firstly instantiated at 23:00. Figure 14 shows screenshots of simulation for both families (the visualizations are produced with [24]).

Att	ID=1	ID=2	ID=1	ID=2	ID=3	ID=4
P	0.6	0.4	0.2	0.2	0.1	0.3
Hy	0.4	0.1	0.3	0.2	0.2	0.1
H	0.3	0.8	0.3	0.0	0.2	0.1
E	0.4	0.5	0.3	0.3	0.2	0.2
M	0.8	0.5	0.6	0.7	0.7	0.7

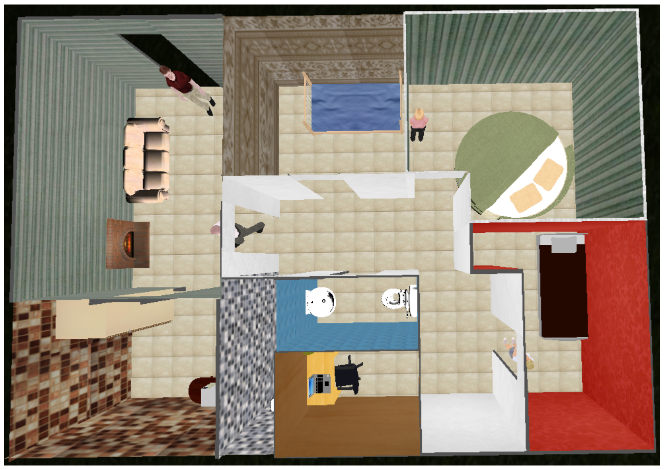
TABLE VI. IN SECOND AND THIRD COLUMNS THERE ARE THE INSTANTIATED ATTRIBUTES FOR A SPECIFIC FAMILY FORMED BY TWO MEMBERS AT 20:00. IN REMAINING COLUMNS THERE ARE THE ATTRIBUTES OF FOUR MEMBERS OF A FAMILY INSTANTIATED AT 23:00.

Now, we consider that the player stop seeing these two families at 24:00 and start again at 10:00 during until 18:00. In this case, the attributes should be created again taking into account the last saved processed time and the standard behaviors. In order to better show the result we illustrate only the evolution of energy attribute for agent ID=1 of family ID=1 in Figure 15.

As the Figure 16 shows, the agents are still living home demonstrating the model's persistence. It is possible to observe a house populated by four people, where an agent leaving to work at morning (a), as well as the agent backing home at night (c). During this period, the agent is out of house and the other three agents keep evolving cohrerently at home(b).



(a)



(b)

Fig. 14. A couple living in a house with one bedroom (a) and a family living in a house with three bedrooms (b).

V. FINAL CONSIDERATIONS

This paper presents a procedural model able to simulate behaviors of virtual agents in indoor environments. Moreover, our approach is able to automatically provide behaviors for virtual agents in order to populate background scenes.

The usage of statistical data allows to provide simulations coherent with virtual world reality. Furthermore, the use of pseudo aleatory numbers makes possible the use of seeds in order to keep the persistence in a procedural way.

Obtained results can be coherently accepted, by visual inspection. As future work, we aim to simulate interactions among agents and objects.

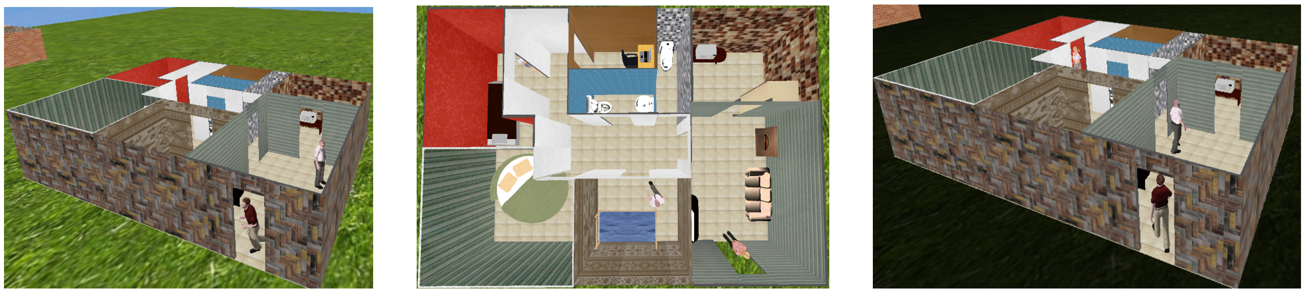


Fig. 16. Three views of user observation from 10:00 to 18:00 on a family composed by 4 people. It is possible to observe an agent ID=1 leaving the home at 10:00 (a); the situation of the house, with only 3 agents at home while agent ID=1 is working (b) and agent ID=1 evolving at home at 18:00 (c).

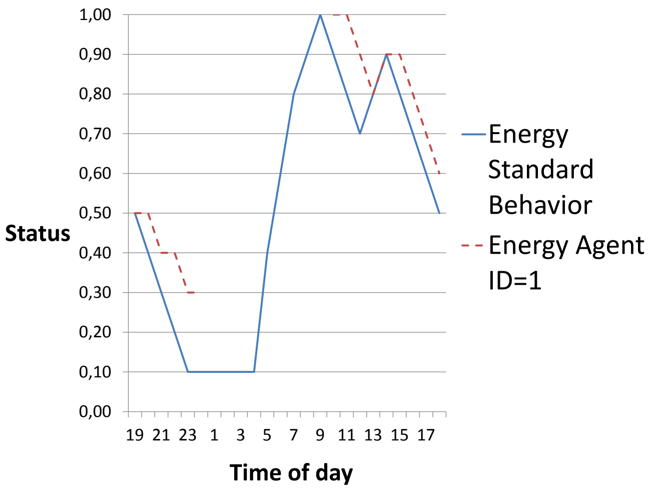


Fig. 15. Attribute energy of agent ID=1 from family ID=1 in function of time, when instantiating (time=19:00), when processing is interrupted since player is not seeing any more the referred family (time=24:00), and when family is re-instantiated at 10:00 during until 18:00.

Acknowledgment

This work was supported by Brazilian research agencies CAPES and FAPERGS.

References

[1] G. T. A. IV, "http://www.rockstargames.com/iv/."

[2] A. Creed, "http://assassinscreed.ubi.com."

[3] W. of Warcraft, "http://us.blizzard.com/pt-br/games/wow/."

[4] M. Fletcher, D.; Yong Yue; Al Kader, "Challenges and perspectives of procedural modelling and effects," in *Information Visualisation (IV), 2010 14th International Conference*, 2010, pp. 543–550.

[5] T. Lechner, B. Watson, U. Wilensky, S. Tisue, M. Felsen, and A. Moddrell, "Procedural modeling of urban land use." in *ACM SIGGRAPH'06.*, 2006.

[6] R. Glass, C. Morkel, and D. Bangay, "Duplicating road patterns in south african informal settlements using procedural techniques," in *Afrigaph '06: Computer graphics, Visualisation and Interaction in Africa, ACM Press*, 2006, pp. 161–169.

[7] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool, "Procedural modeling of buildings," in *ACM SIGGRAPH 2006 Papers*, ser. SIGGRAPH '06. New York, NY, USA: ACM, 2006, pp. 614–623. [Online]. Available: http://doi.acm.org/10.1145/1179352.1141931

[8] M. Larsson, J. Zalzala, and G. Duda, "A procedural ocean toolkit," in *ACM SIGGRAPH 2006 Sketches*, ser. SIGGRAPH '06. New York, NY, USA: ACM, 2006. [Online]. Available: http://doi.acm.org/10.1145/1179849.1179872

[9] F. Marson and S. R. Musse, "Automatic real-time generation of floor plans based on squarified treemaps algorithm." *International Journal of Computer Games Technology.*, vol. 1, p. 10, 2010.

[10] D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, and S. Worley, *Texturing & Modeling: A Procedural Approach*, B. A. Barsky, Ed. Morgan Kaufmann Publishers, 2002.

[11] C. W. Reynolds, "Steering behaviors for autonomous characters," in *Proceedings of the Game Developers Conference*, ser. GDC'99. San Jose, California, USA: Miller Freeman Game Group, 1999, pp. 763–782.

[12] J. Hidalgo, E. Camahort, F. Abad, and M. Vicent, "Procedural graphics model and behavior generation," in *Computational Science ICCS 2008*, ser. Lecture Notes in Computer Science, M. Bubak, G. Albada, J. Dongarra, and P. Sloot, Eds. Springer Berlin Heidelberg, 2008, vol. 5102, pp. 106–115. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-69387-1_12

[13] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dep. of Computer Science, Iowa State University, Tech. Rep. TR98-11, October 1998.

[14] M. G. Choi, J. Lee, and S. Y. Shin, "Planning biped locomotion using motion capture data and probabilistic roadmaps," *ACM Trans. Graph.*, vol. 22, no. 2, pp. 182–203, 2003.

[15] R. A. Metoyer and J. K. Hodgins, "Reactive pedestrian path following from examples," *The Visual Computer*, vol. 20, no. 10, pp. 635–649, 2004.

[16] S. Rodrguez, J.-M. Lien, and N. M. Amato, "A framework for planning motion in environments with moving obstacles," in *IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems*, November 2007, pp. 3309–3314.

[17] M. Kallmann and M. Mataric, "Motion planning using dynamic roadmaps," pp. 4399–4404, 2004, proceedings of IEEE International Conference on Robotics and Automation (ICRA).

[18] A. Kamphuis and M. H. Overmars, "Finding paths for coherent groups using clearance," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2004, pp. 19–28.

[19] J.-M. Lien, S. Rodrguez, J.-P. Malric, and N. M. Amato, "Shepherding behaviors with multiple shepherds," in *Proceedings of the IEEE Inter. Conf. on Robotics and Automation*, April 2005, pp. 3402–3407.

[20] I. Karamouzas and M. Overmars, "Simulating the local behaviour of small pedestrian groups," in *Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology*, ser. VRST '10. New York, NY, USA: ACM, 2010, pp. 183–190. [Online]. Available: http://doi.acm.org/10.1145/1889863.1889906

[21] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[22] V. J. Cassol, F. P. Marson, M. Vendramini, M. Paravisi, A. L. Bicho, C. R. Jung, and S. R. Musse, "Simulation of autonomous agents using

terrain reasoning," in *International Conf. on Computer Graphics and Imaging*, Innsbruck, Austria, 2011.

[23] A. de Lima Bicho, "Da modelagem de plantas dinmica de multides: um modelo de animao comportamental bio-inspirado," Ph.D. dissertation, Universidade Estadual de Campinas - UNICAMP, 2009.

[24] H. Braun, V. J. Cassol, R. Hocevar, F. P. Marson, and S. R. Musse, "Crowdvis: a framework for real time crowd visualization," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ser. SAC '13. New York, NY, USA: ACM, 2013, pp. 989–995. [Online]. Available: http://doi.acm.org/10.1145/2480362.2480551