A 3D Simulator for Intelligent Environment Experiments

Yannick Francillette, Eric Boucher, Sébastien Gaboury and Abdenour Bouzouane
Université du Quebec à Chicoutimi
LIARA
555, boulevard de l'université
Chicoutimi (Qc), Canada G7H 2B1
Email: (yannick.francillette1, eric.boucher1, sebastien_gaboury, abdenour_bouzouane)@uqac.ca

***Abstract*—The advances in sensor networks, electronics and ambient intelligence make creation of intelligent environments (IEs) possible. However, on account of economic and logistic issues the implementation of physical IEs is difficult in research domain. That makes it harder for researchers to experiment new approaches in IE domain.**

In this article, we propose a simulator to build virtual IEs. Simulators are a good alternative to physical IEs. Indeed, virtual IEs does not require expensive resources. Moreover, researchers and designers can conduct experiments anytime and repeat scenarios easily.

Our simulator provides users with a set of virtual sensors and actuators. Our virtual sensors try to reproduce behavior of physical sensors and to produce datasets with the same properties as those generated by real sensors. Our proposition contains a tool to build a home from scratch and a model to define scenarios and behaviors of occupants. It also proposes an interface to control occupants directly. Virtual sensors collect data and generate datasets. Scientists and designers can use these datasets to evaluate and design new approaches in IE domain.

I. Introduction

Intelligent environments (IE) can enhance humans' quality of life and societies significantly. Indeed, they can assist people in some daily activities in order to increase comfort and efficiency. These systems realize the vision of Mark Weiser [1]. Computing is everywhere in service of people instead of being represented by one computer in one room and used by only a few people. In general ways, IEs use several methods to collect and analyze pertinent data in order to provide an appropriated behavior. Consequently, researchers aim to discover models in order to: (i) limit the requirement of expensive sensors; (ii) increase computation speed, efficiency, robustness and flexibility.

Development of new approaches requires access to a dataset. It is possible to use datasets that come from previous implementations and experiments (CASAS for example [2]). However, in some cases, generation and usage of new datasets is required. Unfortunately, it is very difficult to generate new datasets for several reasons: (i) physical IEs are very expensive; (ii) acquisition of data is a time-consuming process; (iii) a lack of suitable participants can make it difficult to conduct some experiments.

Using of a virtual IE is one solution to generate datasets easily and quickly. Virtual IEs provides researcher with serveral advantages [3]. For example, buying of physical sensors and recruiting of suitable participants are not required. Moreover, it is easy and fast to act on environments' configuration (change size of rooms or list of sensors and actuators) and subjects' behavior. In this article, we present a simulator which aims to help IE designers in their building process.

II. Proposition

A. General overview

Our proposition is composed of three layers. Figure 1 shows the general model. Layer named ''Databases" contains two databases. The first one contains datasets generated by simulation and the second one is used as an interface between external program and our virtual actuators. Layer named ''Virtual world" refers to all models related to environment's behavior. Finally, layer named ''Scenarios" refers to all tools provided by our proposition to implement and execute scenarios.

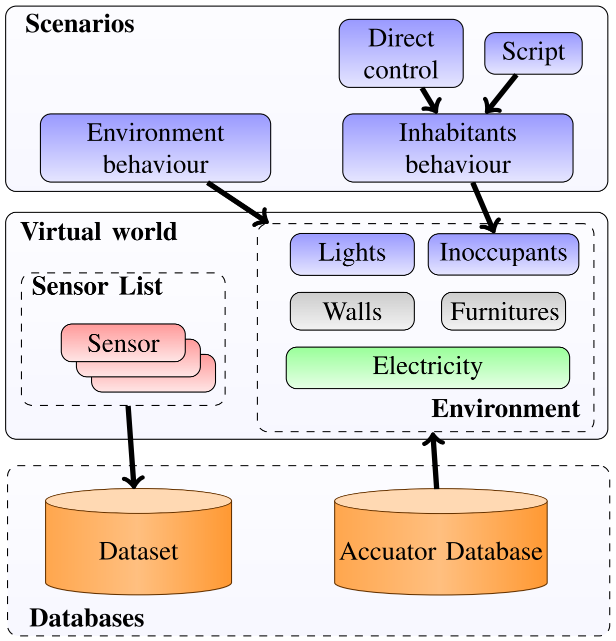

Fig. 1: Scheme giving an overview of the system.

B. Sensors and accuators

Our simulator includes several categories of sensors. This collection allows users to conduct experiments with different configurations of sensors. Table I resumes the set of sensors included in our solution.

Sensors list		
Type	**Value**	**Description**
RFID	Numeric	Strenght of the signal between the antena and the RFID tag
Pressure plate PIR motion Contact Flow meter	Boolean	*true* if an entity pressures *true* if an entity moves in its range *true* the two part are connected *true* if an faucet is opened
Electricity	Numeric	Active and reactive power on a line
Ultrasound Infrared	Numeric	Distance where the beam is stopped

TABLE I: List of sensors that are included in our simulator.

Our proposition uses the following models to simulate sensors' behavior. Contact sensors can be put on furniture such as drawers or doors. This kind of sensor returns current state of furniture linked with it. Flow meters act the same way. The only difference is that flow meter must be put on a faucet. Pressure plates return ''true" when an entity collides with its collision box. Ultrasound and infrared sensors draw a raycast from their position, with an angle of 90 degrees and a length l. If the raycast is cut, the sensor returns distance where it is cut. RFID systems and PIR motion sensors use collider boxes and raycasts. When an entity collides with a box, a raycast is drawn between the sensor and the entity. If nothing intercept the raycast, PIR motion sensor return "true", RFID antenna computes the signal strength. We use the following formula to compute this value:

$$RSSI(x) = (-9,1333ln(x) - 10.726) \times (1 - cos(a)) + noise(s)$$

Where x and a are respectively the distance and the angle between the antenna and the RFID tag. $noise()$ is a function to generate noise and s is the seed of this function.

C. Inoccupant's interactions

Once intelligent environment defined, occupants must interact with it in order to generate data. Our proposition provides users with two modes to control occupants' interactions. The first one is named ''*direct control*" and allows users to control an occupant through keyboard and mouse. Users control occupant's walk and interact with any furniture. The second mode uses scripts to perform different scenarios. Occupants follow and perform activities defined in scripts. Once script is defined, no additional user action is required during the simulation.

We use the behavior tree model to define scripts. Behavior tree is a formalism that is used in planning [4], [5], [6]. In this model, the main goal of an entity is divided in sub activities. The entity has to complete these activities in order to reach its goal. An activity is defined by the couple "objective" and "behavior". Basically, in an activity several actions are done in order to reach or maintain a particular state. This specific state can be an environment state. For example, a door is locked or a light is on. It can be an entity's internal state. For example, in our context, occupant must not be hungry. The term "behavior" refers to behaviors the entity adopts in order to reach the objective.

The model uses a tree structure to organize the activities. In this tree, leaves represent activities. Other nodes are operators. They define how the tree evolve according to current state of activities. Indeed, at each time, activities are in one of the following states:

- **Not running**: activity has not started yet.
- **Running**: activity has started but is not finished yet.
- **Succeeded**: objective of the activity has been reached.
- **Failed**: objective of the activity has not been reached and it is impossible to reach it in the future.

''Succeeded' and ''Failed" are the two states that trigger tree evolution. These states are called "ending state". In order to return current state, each activity contains a function that checks if objective is reached or not.

Activity tree evolves according to ending states of activities and type of operators. Several operators with different operational semantic can be added according to needs. However, we propose the following operators because they are the most common in behavior tree implementations:

- **Sequence**: it executes node sequentially. It starts from the first one and each time a node ends in succeeded state it starts the next one. If a child ends in failed state, it ends in a failed state.
- **Selector**: it executes node sequentially until one ends in succeeded state. In this case, it ends in succeeded state. If all child are in failed state, it ends in failed state.
- **Repeat**: it execute its only children until this one ends in succeeded state.
- **Parrallel**: it executes node parallelly. If a child ends in failed state, it ends in a failed state.

In our solution, behavior tree can be put on occupants or environment's entities. For example, a behavior tree can be put on a phone in order to program a call or on a coffee machine in order to define machine cycle.

D. Implementation

We use Unity game engine to build our simulator. This project is available for download on https://github.com/Iannyck/shima. Our project uses an SQLite for our databases.

III. Example

We propose a scenario example to show datasets generated by our simulator. In this example, we propose smart home. Figure 2 shows a picture of this home architecture. Table II shows list of sensors implemented in this environment.

We have implemented two scenarios in that virtual smart home. In the first scenario, the inhabitant has perform the following activities:

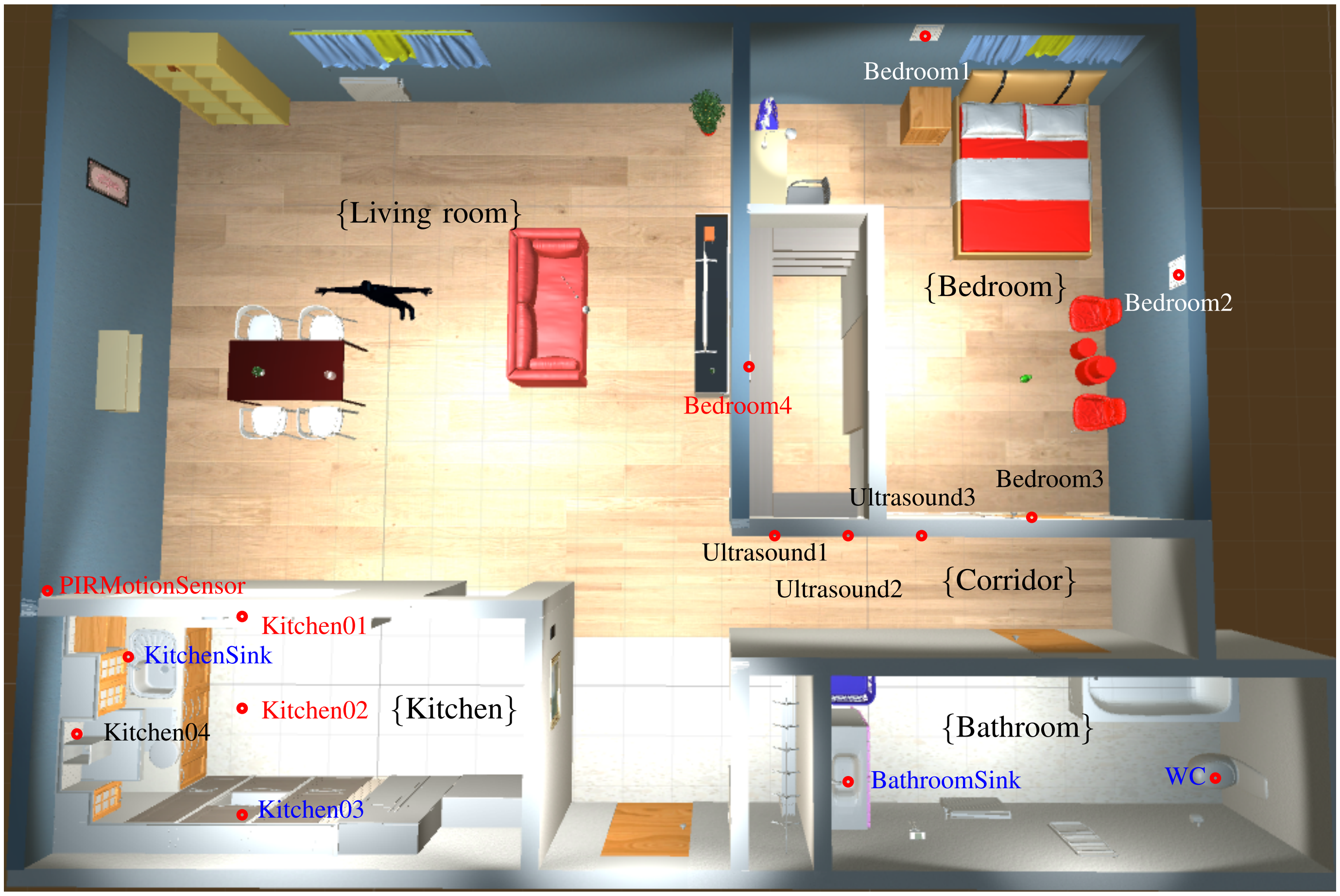

Fig. 2: Screenshot of our smart home architecture.

Sensors list of our example by rooms		
Room	**Sensors**	**ID**
Kitchen	RFID	Kitchen01
	RFID	Kitchen03
	RFID	Kitchen04
	Flow meter	KitchenSink
Kitchen (at ceiling)	RFID	Kitchen02
Bedroom	RFID	Bedroom1
	RFID	Bedroom2
	RFID	Bedroom3
	RFID	Bedroom4
Corridor	Ultrasound	Ultrasound1
	Ultrasound	Ultrasound2
	Ultrasound	Ultrasound3
Living room	PIR Motion	PIRMotionSensor
Bathroom	Flow meter	BathroomSink
	Flow meter	WC

TABLE II: List of sensors in our example. Figure 2 shows position of sensors.

- **Make coffee**: the inhabitant uses the machine to make a cup of coffee.
- **Cook eggs**: the inhabitant uses a frying pan and the cooker to cook eggs.
- **Wash dishes**: the inhabitant washes each plates that are in the kitchen sink.
- **Dress the table**: the inhabitant takes his meal a fork and his beverage a put them to the table in the living room.
- **Undress the table**: the inhabitant takes the dishes on the living room's table and put them into the kitchen sink.

In the second scenario, the inhabitant has to flush the toilet basin then use a detergent to clean the bathroom sink.

In the cooking scenario, the inhabitant mainly triggers the RFID antenna (he moves several items in the kitchen), the movement detector, the flowmeter and he generates changes into the electricity consumption. In the bathroom scenario, he triggers the ultrasound sensor into the corridor, the contact sensor which is on the bathroom door and the flowmeter. Tables III, IV, V and VI shows datasets generated by these scenarios.

IV. Related works

Some IE simulators have been proposed in order to support researchers. SIMACT provides users with an interface to design smart homes and scenarios [7]. In scenarios, user has to detail the sequence of steps involved in the realization of activities. The simulation use Java and run into a 3D environment designed with SketchUp. The simulator store data

Timestamp	Phase	Active power	Reactive power
18:50:39:369	Phase 1	98	0
18:50:39:904	Phase 1	98	0
18:50:40:620	Phase 1	98	0
18:50:40:226	Phase 1	897	0
18:50:40:404	Phase 1	897	0
18:50:40:941	Phase 1	897	0
18:50:41:105	Phase 1	897	0

TABLE III: Sample of the electricity dataset generated by our example. This sample only shows phase 1. In our example, reactive power is always equal to 0 because of our electronic device settings.

Timestamp	Sensor ID	Signal (DB)	Tag
18:50:19:743	Kitchen01	-44.21692	TagSpoon
18:50:19:743	Kitchen01	-44.1373	TagMug
18:50:20:626	Kitchen04	-43.40419	TagMug
18:50:20:626	Kitchen04	-51.76328	TagFrypan
18:50:20:970	Kitchen02	-50.53626	TagFrypan
18:50:20:970	Kitchen02	-41.93156	TagSpoon

TABLE IV: Sample of the RFID dataset generated by our example.

Timestamp	Sensor ID	Beam
18:38:46:447	Ultrasound 3	4.108222
18:38:46:868	Ultrasound 3	3.783601
18:38:47:317	Ultrasound 3	3.513622
18:38:47:837	Ultrasound 2	2.883075
18:38:48:601	Ultrasound 1	3.537868

TABLE V: Sample of the ultrasound dataset generated by our example.

Timestamp	Sensor ID	Type	Value
18:38:55	Flow meter	WC	*true*
18:51:43:229	PIRMotionSensor	PIR Motion Sensor	*true*
18:51:44:465	PIRMotionSensor	PIR Motion Sensor	*true*
18:51:49:532	PIRMotionSensor	PIR Motion Sensor	*true*
18:51:50:825	PIRMotionSensor	PIR Motion Sensor	*true*

TABLE VI: Sample of the binary dataset generated by our example.

about objects interaction into a database. However, SIMACT does not really simulate physical behavior as sensors, it detects when an action is performed on an object and store the contact information.

PerSim 3D is an enhancement of PerSim 1.5 proposed in [8]. It proposes several categories of sensors that can be selected and set into the virtual smart home. PerSim also allows the user to design the architecture of the home. Like in SIMACT, the user can play and see the simulation through a 3D rendering of the scene. PerSim provides the user with more kinds of sensors than SIMACT, however, PerSim seems not to record simulated sensors raw data. In fact, sensor data is transformed in order to provide the fact that it is activated or not.

In [9], authors propose a 2D simulator. Authors slip their sensors in two categories: motion and non-motion. The non-motion sensors are like environment variable. They are increased while the inhabitant do an action where they are involved. For example, the water consumption is increased when the inhabitant use a faucet. The motion sensors are sensors that detect if the inhabitant is in a particular zone. Then it computes a distance in order to simulate a signal strength, for example. This simulator works in a 2D world and it does not provide a real time rendering.

Unlike, some solutions in related works our solution to generate raw datasets that are similar to the real dataset. It proposes several kinds of sensors and allows the creation of virtual intelligent environment from scratch. Moreover, users can act on entities' behavior.

V. Conclusion and future works

In this paper, we have presented our IE simulator. This simulator allows IE researcher to generate datasets in order to evaluate their contributions. Moreover, it allows IE design to evaluate their IE proposition before the implementation of this one. Our simulator allows the building of the environment (position of walls, doors, windows and selection of items) and the selection and positioning of sensors. It proposes a selection beside several kinds of sensors, RFID, ultrasound and infrared, pressure plate, contact sensors, flowmeter, movement detector. These sensors generate data similar to those generated by physical sensors. Our simulator proposes two modes to generate datasets. The first one is an interactive mode where the user control an occupant and can interact with items in the home. The second mode is a scenario mode where occupants are controlled by scripts.

References

[1] M. Weiser, "The computer for the 21st century," *Scientific American*, no. Communications, Computers, and Network, septembre 1991. [Online]. Available: http://wiki.daimi.au.dk/pca/_files/weiser-orig.pdf

[2] D. J. Cook and L. B. Holder, "Sensor selection to support practical use of health-monitoring smart environments," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 4, pp. 339–351, 2011.

[3] J. Synnott, C. Nugent, and P. Jeffers, "Simulation of smart home activity datasets," *Sensors*, vol. 15, no. 6, pp. 14 162–14 179, 2015.

[4] P. Ogren, "Increasing modularity of uav control systems using computer game behavior trees," in *AIAA Guidance, Navigation and Control Conference, Minneapolis, MN*, 2012.

[5] A. J. Champandard, M. Dawe, and D. Hernandez-Cerpa, "Behavior trees: Three ways of cultivating game ai," in *Game Developers Conference, AI Summit*, 2010.

[6] A. Marzinotto, M. Colledanchise, C. Smith, and P. Ogren, "Towards a unified behavior trees framework for robot control," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5420–5427.

[7] K. Bouchard, A. Ajroud, B. Bouchard, and A. Bouzouane, "Simact: A 3d open source smart home simulator for activity recognition with open database and visual editor," *International Journal of Hybrid Information Technology*, vol. 5, no. 3, pp. 13–32, 2012.

[8] A. Helal, K. Cho, W. Lee, Y. Sung, J. Lee, and E. Kim, "3d modeling and simulation of human activities in smart spaces," in *Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2012 9th International Conference on*. IEEE, 2012, pp. 112–119.

[9] B. Kormányos and B. Pataki, "Multilevel simulation of daily activities: Why and how?" in *Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–6.