EmohawkVille: Virtual City for Everyone

David Holaň and **Jakub Gemrot** and **Martin Černý** and **Cyril Brom** [1]

Abstract. Despite recent progress, behavior of non-player characters (NPCs) in contemporary games is still kept rather simple. This is an opportunity for the academia to develop novel techniques and tools that would allow for easier creation of complex behaviors that are resilient to the dynamicity implied by the presence of the player. There already exist languages within multiagent community that are thought to be suitable for NPC behaviors creation, but they are usually tested in simplistic environments and our experience indicates that applying them to complex 3D worlds introduces significant obstacles. This is part of the reason why simple reactive techniques are prevalent in game industry practice. Moreover there is no publicly available research-friendly 3D virtual world with sufficient complexity that would allow developers to evaluate their languages and tools in a more realistic setting and improve them toward practical applicability. In this demo we present EmohawkVille: an open-source first-person 3D virtual world that is a candidate for such an environment.

1 Introduction

Many contemporary computer games take a great effort to achieve a high level of believability of their virtual worlds. This is especially true for games with large open worlds, where the user is free to discover the environment on his own and is relatively unconstrained by the game. One of the challenges that arise in this scenario is the problem of choosing the right higher-level action for the NPCs (e.g., move to a point, pick up an item, use an item, ...). Since the game industry relies almost exclusively on simple reactive techniques which make creation of complex behaviors rather time-consuming and costly, non-player characters (NPCs) display complex behaviors only during crucial game events. In between, the NPC behaviors are schematic at best.

The main issue is that going beyond simple behavior and still maintaining the suspension of disbelief introduces significant difficulties to the NPC behavior authoring. There are many possible obstacles to NPC goals and if they are not taken into account, the NPCs are easy for the player to "break" and may provide even worse illusion of a real world than rather static NPCs.

For a truly alive open world, dozens of different and often complex scenarios are needed, which implies that the world needs to be equipped with a rich ontology of items and actions NPCs (as well as the player) can perform.

As the world ontology grows, the number of meaningful NPC action sequences increases and the behavior complexity rises. Not only the means-ends analysis becomes more demanding, new problems emerge such as transitional behaviors, joint behaviors, behaviors ordering or behaviors interleaving [6]. At the same time, game studios usually cannot afford to let an expert AI programmer design such day-to-day behaviors, because that would be cost-prohibitive. Most of the NPC design is thus usually carried out with the aid of some visual tool by scripters with little programming experience.

At this place, academia could provide action selection mechanisms (ASM) and accompanying tools that would help inexpert scripters to create complex behaviors that are *interactively believable*, that is, behaviors that sustain their believability under nondeterminism brought by the player. However, most of the academic research is carried out in environments that either have simple ontologies or are static or discrete. Games on the other hand are dynamic, multi-agent environments that can be for all practical purposes considered continuous in both time and space. There are languages and techniques that can be applied to such worlds: either from the multiagent community or the field of robotics or automated planning. However, to our knowledge, there is currently no 3D virtual world publicly available that would provide rich ontology for NPCs out of the box. This means that in this particular problem area, academia is one step behind the industry — we do not even have an environment to work with.

Note that raw frameworks such as Unity [7] are not sufficient as creating a rich world in a raw framework is a substantial amount of work. An important part of the environment is also the possibility to develop the NPC behavior with a high-level language such as Java since nearly all agent languages of interest can be invoked from Java code. We are not aware of any complex 3D environment that would meet all those requirements. See our paper [4] for a thorough comparison of possible candidates.

Previous research has shown that applying agent languages to 3D environments is neither straightforward nor guaranteed to yield better results than using a general programming language [2, 5]. Common issues with agent languages are incomplete debugging and tool support, some of the architectures are also hard to debug in principle (e.g., because of inherent paralellism). Many agent languages are also declarative in nature, while game worlds feature lots of mechanics that are hard to express declaratively (e.g., determining which object is hit by an arrow). Proper evaluation of agent languages is thus critical.

In this demo, we present an extension of the Pogamut 3 platform [3] called EmohawkVille, the first step towards an open-sourced complex simulation of NPC everyday life in 3D virtual world. We believe that creating a fully working, accessible and polished environment fosters academic progress. The large amount of research work evaluated on Pogamut for Unreal Tournament 2004 supports this view. We have also exerted great effort to make EmohawkVille a mature tool. In practice, there is a long chain of components that are needed to fully connect high-level AI with an NPC: sensors and actuators interface, navigation and pathfinding, character animation support are among the most important, but the list is far from exhaus-

[1] Charles University in Prague, Czech Republic email: {paladin.invictus,jakub.gemrot,cerny.m}@gmail.com, brom@ksvi.mff.cuni.cz

tive. In EmohawkVille, we have resolved large part of those issues on behalf of the researcher. The quality of the EmohawkVille environment was evaluated in a small-scale user study and by use of the environment in our teaching curriculum.

2 General Description

EmohawkVille is a first-person virtual world with detailed interactive elements of day-to-day life. There is a general framework that supports interaction with items, continual actions and processes and inter-agent communication including trade. There is a set of ready-made assets for a cooking scenario. For example, an agent or a human player can pick up a piece of meat, put it on a chopping board and slice it and then fry the slices on a pan (charring the food if he does not add oil or forgets to flip the meat). The cooking scenario was chosen as our first because it features plethora of complex procedures yet it is easy to grasp by programmers and non-programmers alike and is gender-neutral.

EmohawkVille is based on Unreal Development Kit (UDK) [1] and thus is capable of displaying the world in state-of-the-art graphics. UDK is free for educational and non-commercial use and EmohawkVille itself is available under GPLv3[2].

In EmohawkVille the world mechanics are implemented in UnrealScript - a proprietary language deployed with the UDK toolkit. The Pogamut platform provides a high-level Java interface to the UDK for writing the actual AI and takes care of many common tasks (pathfinding with A* and smooth path following, caching sensory data to a blackboard, etc.). Both the UDK and the Java part have been designed with possible further extensions in mind and the basic NPC support is separated from the model of the general EmohawkVille ontology, which is in turn separated from the implementation of the specific mechanics for our cooking scenario. The UDK part also fully supports interaction with a human user through the UDK visual client.

At this moment, EmohawkVille features 20 item types (food, cutlery, cooking tools, ...) and a cooking stove (part of the environment). Interaction is provided by 14 actions, of which nine are instant and four initiate a longer-lasting process, e.g., chopping a vegetable or stirring a broth. An overview of the available items is visible in Figure 1.

Figure 1. A screenshot of the environment.

The central complexity of the NPC behavior stems from the simulation of cooking. Some ingredients can be boiled, some fried. The speed of cooking is determined by the temperature of respective stoves. Water evaporates from pots and ingredients may burn or char if not stirred or flipped in the pot or the pan. The cooking theme provides important challenges to the NPC behavior creation: cooking a meal may require a long sequence of actions (more than 20), effectivity is increased by performing processes in parallel possibly requiring cooperation of multiple chefs, the player may both support and sabotage the cooking NPC.

Every aspect of the environment and the agents is programmable. EmohawkVille is ready for a researcher to plugin any high-level decision making mechanism (planning, machine learning, ...) without the need to handle low-level details. More detail of the environment is given in our paper [4].

3 Demo Presentation

In our demo presentation we would like to show the environment and its richness, let the spectators interact with the environment themselves, helping a preprogrammed agent to cook a complex meal or sabotaging his effort. We would also like to show that programming the behaviors is easy and EmohawkVille thus lets the researcher focus on the action selection exclusively. This will be demonstrated by a live creation of a cooking NPC and we would enable hands-on programming experience to the spectators.

A video presentation of the environment may be found at `http://www.youtube.com/watch?v=G7lKXkR2Xgg`

ACKNOWLEDGEMENTS

This research was supported by SVV project number 260 224.

REFERENCES

[1] Epic Games Inc. Unreal development kit documentation. `http://www.unrealengine.com/udk/documentation/`, 2009. Last checked: 2015-03-30.

[2] Jakub Gemrot, Zdeněk Hlávka, and Cyril Brom, 'Does high-level behavior specification tool make production of virtual agent behaviors better?', in *Proceedings of CAVE'12*, pp. 167–183, Berlin, Heidelberg, (2013). Springer-Verlag.

[3] Jakub Gemrot, Rudolf Kadlec, Michal Bída, Ondřej Burkert, Radek Píbil, Jan Havlíček, Lukáš Zemčák, Juraj Šimlovič, Radim Vansa, Michal Štolba, Tomáš Plch, and Cyril Brom, 'Pogamut 3 can assist developers in building ai (not only) for their videogame agents', in *Agents for Games and Simulations*, eds., Frank Dignum, Jeff Bradshaw, Barry Silverman, and Willem Doesburg, LNCS 5920, 1–15, Springer-Verlag, (2009).

[4] Gemrot J. Černý M. Holaň, D., 'Emohawkville: Towards complex dynamic virtual worlds', in *Proceedings of GAMEON'2013*, pp. 52–58, (2013).

[5] Radek Píbil, Peter Novák, Cyril Brom, and Jakub Gemrot, 'Notes on pragmatic agent-programming with Jason', in *Programming Multi-Agent Systems*, LNCS 7217, 58–73, Springer, (2012).

[6] Tom Plch, *Action selection for an animat*, Master's thesis, Charles University in Prague, 2009.

[7] Unity Technologies. Unity documentation. `http://docs.unity3d.com/Documentation/Manual/`, 2005. Last checked: 2015-03-30.

[2] EmohawkVille may be downloaded from `http://pogamut.cuni.cz/main/tiki-index.php?page=EmohawkVille`