Pogamut 3 Can Assist Developers in Building AI for Their Videogame Agents

Jakub Gemrot, Rudolf Kadlec, Michal Bída, Ondřej Burkert, Radek Píbil, Jan Havlíček, Lukáš Zemčák, Juraj Šimlovič, Radim Vansa, Michal Štolba, Cyril Brom

Charles University in Prague, Faculty of Mathematics and Physics
Malostranské nám. 2/25, Prague, Czech Republic
{jakub.gemrot, rudolf.kadlec}@gmail.com, brom@ksvi.mff.cuni.cz

Abstract. Many research projects oriented on control mechanisms of virtual agents in videogames have emerged in last years. However, this boost has not been accompanied with the emergence of toolkits supporting development of these projects, which slows down the progress of the field. Here, we present Pogamut 3, an open source platform for rapid development of behaviour of virtual agents embodied in a 3D environment of the Unreal Tournament 2004 videogame. Pogamut 3 is tailored to support research as well as educational projects.

Keywords: Virtual agents, behaviour, control mechanisms, Unreal Tournament, 3D environment, agent development toolkit.

1 Introduction

The development of control mechanisms of videogame agents (that is, their “artificial intelligence”) is hard by itself. The life of developers is typically complicated further by the fact that they have to solve many tedious technical issues that are out of their main scope. For instance, they have to integrate their agent with a particular 3D environment, build at least a simple debugging tool or write a code for low-level movement of the agent. This list continues for pages and developers address its items again and again, a waste of energy.

For last three years, we have been developing the Pogamut 3 toolkit, which provides general solutions for many of these issues, allowing developers to focus on their main goals. Importantly, Pogamut 3 is designed not only for advanced researchers, but also for newcomers. Pogamut 3 can help them to build their first virtual agents, a feature which makes it applicable as a toolkit for training in university and high-school courses [5].

The purpose of this paper is to review main features of Pogamut 3, its architecture, and work in progress, providing supplementary information for the demonstration of Pogamut 3 at the AAMAS-09 Workshop on Agents for Games and Simulations.

2 Features of Pogamut 3

The agent development cycle can be conceived as having five stages: (1) inventing the agent, (2) implementing the agent, (3) debugging the implemented agent, (4) tuning the parameters of the agent, and (5) validating the agent through series of experiments. Individual features of Pogamut 3 were purposely designed to provide the support during the latter four stages.

The features of Pogamut 3 includes:

1) a binding to the virtual world of the Unreal Tournament 2004 videogame (UT2004) [7],
2) an integrated development environment (IDE) with a support for debugging,
3) a library with sensory-motor primitives, path-finding algorithms, and a support for shooting behaviour and weapon handling,
4) a connection to the POSH [6], which is a reactive planner for controlling behaviour of agents, and a visual editor of POSH plans (the editor is not included in the current release yet),
5) a support for running experiments, including distributed experiments run on a GRID.

Integral part of the Pogamut 3 toolkit is the homepage [3] where one can download the installer, see tutorial videos, learn more about the platform, and get support on forums.

3 Details of the Features and Technicalities

The Pogamut 3 toolkit integrates five main components: *UT2004*, *GameBots2004*, the *GaviaLib library*, the *Pogamut agent*, and the *IDE* (Fig. 1). These components implement the abovementioned features, enabling a user to create, debug, and evaluate his or her agents conveniently.

UT2004 is a well known action videogame with so-called “partly opened” code. In particular, the game features UnrealScript, a native scripting language developed by the authors of UT2004, in which a substantial part of the game is programmed: almost everything except of the graphical and the physical engine. The code in UnrealScript can be modified by users. Additionally, the game features a lot of pre-built objects and maps, and also a level editor. These two points allow users to create new game extensions and blend them with the original game content. In the context of the Pogamut 3 toolkit, the game provides the virtual environment for running agents.

One can implement agents directly in UnrealScript; however, it is often advantageous, both for educational as well as experimental reasons, to develop the agents using another, external, mechanism. This means that one has to create a binding to the UT2004. Several years ago, Adobbati et al. developed a generic binding called GameBots [1] for this purpose, which were used by many from then. The original GameBots worked with UT99. GameBots2004 is our extension of the original GameBots.

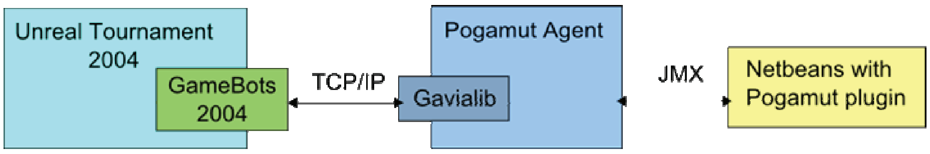
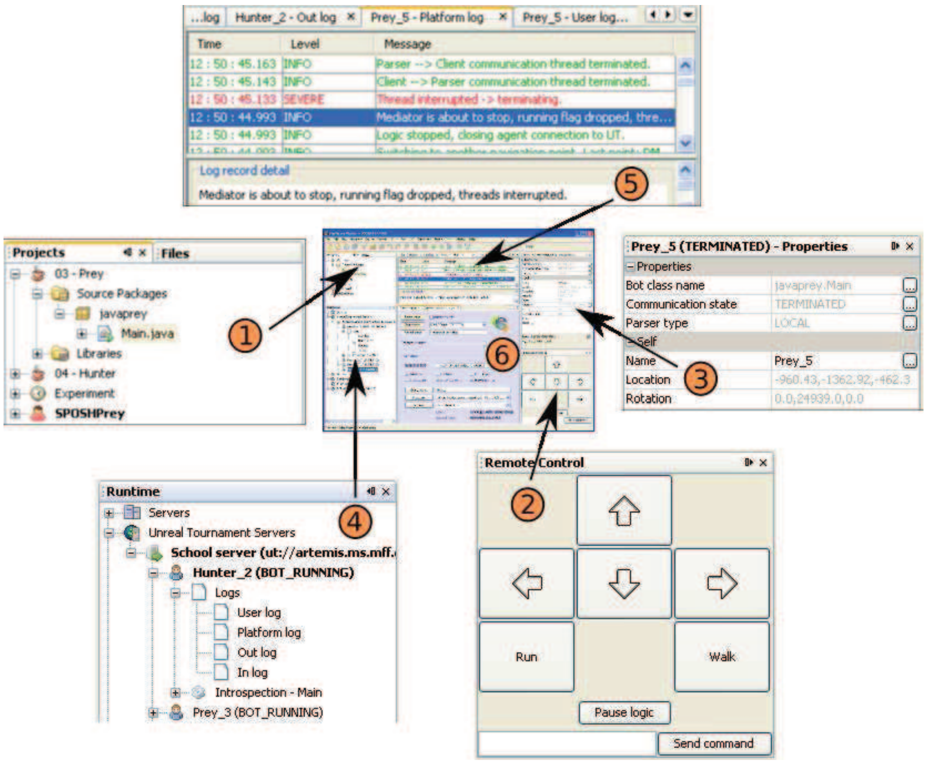


Fig. 1. Pogamut architecture.



```
Time             Level    Message
12 : 50 : 45.163 INFO     Parser --> Client communication thread terminated.
12 : 50 : 45.143 INFO     Client --> Parser communication thread terminated.
12 : 50 : 45.133 SEVERE   Thread interrupted -> terminating.
12 : 50 : 44.993 INFO     Mediator is about to stop, running flag dropped, thre...
12 : 50 : 44.993 INFO     Logic stopped, closing agent connection to UT.
```

Mediator is about to stop, running flag dropped, threads interrupted.

Prey_5 (TERMINATED) - Properties

Properties	
Bot class name	javaprey.Main
Communication state	TERMINATED
Parser type	LOCAL
Self	
Name	Prey_5
Location	-960.43,-1362.92,-462.3
Rotation	0.0,24939.0,0.0

Fig. 2. Pogamut 3 NetBeans plugin (6) and its parts (1-5). The parts are described in the text.

GameBots2004 exports information about the game environment through TCP/IP text based protocol allowing a user to connect to the UT in the client-server manner. This means that the user implements an agent's control mechanisms in the client and uses GameBots2004 as the server. Note, that GameBots2004 can be used separately without other components of Pogamut 3.

The GaviaLib library is a general purpose Java library for connecting agents to almost any virtual environment. Only mild assumptions have been imposed upon virtual environments; in a nutshell, an environment has to work with objects and be able to provide information in an event-based manner. GaviaLib handles communication with the environment, manages object identities, notifies the agent of important object events (e.g. object appeared/disappeared/was changed) and supports out of the box remote control of agents through JMX [11], which is a standard Java protocol for remote instrumentation of programs. The GaviaLib's layered architecture makes it possible to use different means of communication with the virtual world. A GaviaLib agent can be connected through plain text TCP/IP protocols like GameBots2004 are, through CORBA or, in the case of Java environments, it can reside in the same Java Virtual Machine as the world simulator and communicate directly through method calls. Presently, the only binding provided with GaviaLib is the UT2004 binding, which employs Gamebots2004, as detailed below.

The Pogamut agent is a set of Java classes and interfaces built on top of the GaviaLib library. It adds UT2004 specific features, such as UT2004's sensory-motor primitives, the navigation using the UT2004 system of way-points and auxiliary classes providing information about the game rules. Developers can program UT2004 agents using the classes of the Pogamut agent; they can implement an agent's control mechanism directly in Java or use the reactive planner POSH. POSH has been developed as an independent action selection planner by Joanna Bryson [6] and Pogamut 3 exploits it as a plug-in. This planner enforces the design that clearly separates low level actions and sensory primitives (coded in Java or Python) from a high level hierarchical plan of an agent's behaviour.

The IDE is developed as a NetBeans plugin [12] (Fig. 2). It can communicate with running agents via JMX. The IDE offers a designer many features, such as empty agent project templates, example agents (e.g. hunter and prey agents or a khepera-like agent) (Fig. 2, Window 1), management of UT2004 servers, list of running agents (Fig. 2, Window 4), introspection of variables of agents and a quick access to their general properties (Fig. 2, Window 3), the step-by-step debugging, log viewers (Fig. 2, Window 5), and the manual controller of positions of agents (Fig. 2, Window 2). However, the Pogamut agents can be developed without this plug-in in any other Java IDE.

In general, the components of the Pogamut 3 toolkit can be used *per partes*. As already said, one can use GameBots2004 independently as well as parts of the GaviaLib library or the IDE.

4 Conclusion and Prospect Work in Progress

The Pogamut 3 toolkit provides many general solutions for videogame agents developers. The important facet of Pogamut 3 is that it has been designed not only for advanced researchers, but also for newcomers. Its applicability for beginners has been proved by using it as a toolkit for training in a university and a high-school course on programming virtual agents [5].

The current release of Pogamut has also several limitations. Its integration with UT2004 and the support provided by GaviaLib classes make it most suitable for gaming AI projects; this suits some, but is unsuitable for others. We are now extending Pogamut 3 with features that can be utilised in the context of virtual storytelling and computational cognitive psychology. Most notably, Pogamut 3 is being extended by a storytelling manager, an emotion model based on the ALMA model [8], a support for gestures coded in BML [13], and the ACT-R cognitive architecture [2] for controlling agents that will become an alternative to the POSH planner or the plain Java. We are also working on an educational scenario that will utilise these technologies.

To prove the flexibility of the GaviaLib library, we would like to connect it to a new environment. We are considering the Defcon game [10] or Virtual Battle Space 2 (VBS2) [4]. The latter features the industry standard HLA interface (IEEE 1516) [9]. The creation of a general GaviaLib–HLA bridge would make it possible to connect many more environments with a minimal effort.

Acknowledgments. This work was partially supported by the project CZ.2.17/3.1.00/31162 that is financed by the European Social Fund, the Budget of the Czech Republic and the Municipality of Prague. It was also partially supported by the Program "Information Society" under project 1ET100300517, by the grant 201/09/H057, and by the research project MSM0021620838 of the Ministry of Education of the Czech Republic.

References

1. Adobbati, R., Marshall, A. N., Scholer, A., and Tejada, S.: Gamebots: A 3d virtual world test-bed for multi-agent research. In: Proc. of the 2nd Int. Workshop on Infrastructure for Agents, MAS, and Scalable MAS, Montreal, Canada (2001)
2. Anderson, J. R.: How can the human mind occur in the physical universe? Oxford University Press (2007)
3. "Artificial Minds for Intelligent Systems" Research Group: The Pogamut platform. Charles University in Prague. http://artemis.ms.mff.cuni.cz/pogamut [4.3.2009]
4. Bohemia Interactive: Virtual Battle Space 2. http://virtualbattlespace.vbs2.com/ [4.3.2009]
5. Brom, C., Gemrot, J., Burkert, O., Kadlec, R., Bída, M.: 3D Immersion in Virtual Agents Education. In: Proc. of 1st Joint Int. Conference on Interactive Digital Storytelling ICIDS 2008, LNCS 5334, Erfurt, Germany, Berlin: Springer-Verlag. (2008) 59-70
6. Bryson, J. J.: Intelligence by Design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agents. PhD thesis, Massachusetts Institute of Technology (2001)
7. Epic Games: Unreal Tournament 2004. (2004) URL: http://www.unreal.com [4.3.2009]
8. Gebhard, P.: ALMA – A Layered Model of Affect. In: Proc. of the 4th Int. Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'05), Utrecht. (2005) 29– 36
9. IEEE: IEEE 1516, High Level Architecture (HLA). http://www.ieee.org (2001)
10. Introversion: Defcon. http://www.introversion.co.uk/defcon/ [4.3.2009]
11. Sun Microsystems: JMX. http://java.sun.com/javase/6/docs/technotes/guides/jmx/ [4.3.2009]
12. Sun Microsystems, Netbeans IDE. http://www.netbeans.org [4.3.2009]
13. Vilhjalmsson, H., Cantelmo, N, Cassell, J., Chafai, N., Kipp, M., Kopp, S., et al.: The Behavior Markup Language: Recent Developments and Challenges. In: Proc. of 7th Conference on Intelligent Virtual Agents (IVA 07), LNAI 4722, Springer-Verlag (2007) 99-111