Selecting the Best Open Source 3D Games Engines

Rafaela V. Rocha, Rodrigo V. Rocha, Regina B. Araújo

Federal University of São Carlos
Department of Computer Science, Brazil

Abstract

The creation of a computer game is a complex task and engine games are built to facilitate game development. There are several available engines and each one has different features and purposes. The challenge is to choose the most appropriate for a particular game project. This paper provides a compact and comprehensive overview of the best open source 3D engines and a discussion of the most important features. The evaluated engines that support both Windows and Linux platforms are: Blender Game Engine, Crystal Space, Delta3D, Irrlicht, jMonkey Engine, Ogre3D, OpenSceneGraph and Panda3D.

Keywords: Open Source 3D Game Engine, Blender Game Engine, Crystal Space, Delta3D, Irrlicht, jMonkey Engine, Ogre3D, OpenSceneGraph, Panda3D

Authors' contact:
{rafaela_rocha, rodrigo_rocha}@dc.ufscar.br

1. Introduction

Computer games are software for different purposes, from entertainment to training simulations. Game engines are software systems that enable game development and execution [1,2]. These systems can provide features from communication with input and output devices to physics calculations and artificial intelligence. Generally, the features are grouped into components. A large number of companies that are specialized in building engines or components for games have emerged in the last years. One of the challenges for game developers is to choose the engine that best meets the needs of each project. Different features can be used for choosing the best engine for a game. In the Web there are several forums and articles with different opinions about the best 3D engines. However, for beginners in this area, who do not have the important criteria clearly defined, it can be a complex task to make the best decision. This paper presents a comparison and analysis of the best open source 3D games engines for both Windows and Linux platforms. These engines have documentation, large user community and stable source code available. The main engine features are introduced and the engines are presented and analyzed. The paper is organized as follows: In Section 2, the main features and fundamentals of engines are introduced. In Section 3, the top game engines are presented. In Section 4, a compact view of these engines is provided, followed by Discussions, Conclusions and References.

2. Concepts and Features of Engines

An engine provides an abstraction to developers to make their project development easier by hiding complex concepts and providing powerful resources (usually described as libraries or APIs) [2]. There are specific engines, such as, physics (PhysX, ODE, Newton, Bullet), audio (OpenAL), input (InterSense, PLIB, SDL, OpenInput), GUI (ceGUI, QT, FLTK) and network (GNE, HLA, HawkNL, RakNet, OpenTNL), among others. Several factors can be considered when choosing a suitable engine, such as recent technology, license and distribution costs, source code available, game type to be created, platform, developer community, sound and networking systems support, documentation available, rendering, user interface, character and vehicle animation, collision detection and tools that can be used [2, 3]. In this paper, it is assumed that the game developer is evaluating engines that: (1) have a recent stable version; (2) have no license cost; (3) have the source code available; (4) have support for both Windows and Linux platform; (5) are not only for a specific game type; and (6) have available documentation and forum. These assumptions ensure the maturity and stability of the open source engines and the development support. Next section presents the best game engines under the above criteria.

3. Description of the best Graphics and Game Engines

There are several sources for presenting and comparing 3D engines [1, 3, 4, 5]. Although some of them are pretty good and complete, it is not easy for a beginner to find and decide about the best engine for his/her project. DevMaster.net [6], for instance, contains 303 engine records available for research. Although it has a great engines list, with a complete feature description, some of the engines are outdated and comparing the engines is a complicated task, especially for beginners. This section provides a general and more compact view of some of the most important features of available 3D engines with their structure and limitations when compared to each other. By having the top Google results and the engines listed in the other sources above, and considering the engines that run on both Windows and Linux OS, free license and source code available for download, eight open source game engines were selected for analysis, which are described and compared in the next sections.

3.1 The Blender Game Engine

Blender Game Engine [7] is a system for 3D modeling, animation, compositing, rendering and development of 3D games and simulations in real-time. It consists of several simulation engines such as particles, rigid body dynamics, soft body and fluid, among others. Blender contains a 3D modeling environment with graphical interface and rendering and a game engine [7]. The Blender Game Engine uses logic blocks as a visual way to create interactions within the game. Blender is composed of several external libraries, APIs and operating system libraries.

3.2 Crystal Space

Crystal Space (CS) [8, 9] is a framework for developing 3D graphics applications. The CS rendering engine is modular, configurable and extensible. It supports OpenGL and advanced features such as shadows, lighting systems, physics, animation, foliage, terrain, physical devices, among others. The Crystal Entity Layer (CEL) provides the game engine that allows developers to manage entities (non-visual or visual objects in the game). CELstart is a CEL-based environment that allows the creation of games via scripting (Python or XML script). The use of the game engine (CEL) is optional. If using it, the entities must be declared as specified in CS, losing some flexibility of the models, but with the option of using artificial intelligence and other resources of the engine [9].

3.3 Delta3D

Delta3D is a game and simulation engine developed by the Modeling, Virtual Environment and Simulation (MOVES) Institute at the Naval Postgraduate School [10, 11]. Delta3D integrates and provides different APIs. These APIs provide support for: rendering, particle system, scripting language, network management (an interface to the network compliant HLA standard reference and an integration with internal components of the RTI [12]), animation, among others. Delta3D allows direct access to its APIs as needed. Delta3D provides several tools, such as, the 3D model viewer, the HLA Stealth Viewer, the particle editor, the BSP compiler and the Simulation, Training, and Game Editor (STAGE) [11].

3.4 Irrlicht

Irrlicht [13] is a 3D engine that creates an abstraction to the application by hiding the low level implementation using OpenGL, DirectX or software renderer. It provides the interfaces for system input and output, graphical user interface, video driver access, scene management and character animation. Irrlicht provides free libraries and tools for adding features, such as sound control, XML parser and 3D world as well as level editor.

3.5 jMonkey Engine

jMonkey Engine (jME) [14] is a graphics API written in Java that provides a scene graph architecture. jME has support to LWJGL (Lightweight Java Game Library) and JOGL (Java OpenGL), which allows access to OpenGL using Java language. The OpengGL ES renderer is planned for futures release, well as Android support. JOGL OAL (Open Audio Library) provides audio functionalities for jME. The Scene Graph provides a tree structure that allows organizing the 3D scene objects. jME supports environmental mapping, particle systems, keyframe and skeletal animation, pathfinding algorithms (IA).

3.6 Ogre3D

Object-Oriented Graphics Rendering Engine (OGRE3D) was developed by Steve Streeting and is currently maintained by OGRE Team [15]. It manages the state of automatic processing and transparencies rendering. It supports shaders written in both machine and high-level languages. It also has flexible data formats mesh, animation support, management scene, scene graph hierarchy, different techniques of shadow rendering, special effects, particle system, among other features. In Ogre3D there are many manager classes that control distinct resources. There are also many wrappers that provide features such as networking, collision detection, physics, sound, input control, artificial intelligence and graphical user interface.

3.7 OpenSceneGraph

OpenSceneGraph (OSG) is a high-level graphics toolkit [16]. It allows the creation of high performance graphics applications with high level calls since it provides an object-oriented framework that accesses the OpenGL. It uses a scene graph that manages the objects in virtual space. OSG has the core, the plugins, and the node kits. OSG plugins allow to export, import and load different 3D models. Node kits are separate libraries that provide features such as effects, particle systems, visual simulation, animation, interactive controls, terrain and volume rendering, among others. In the OSG core, there are the scene graph classes, utility classes and database reading and writing libraries.

3.8 Panda 3D

Panda3D is a 3D engine for rendering and game development, developed by Disney Imagineering and currently maintained by Carnegie Mellon University's Entertainment Technology Center (CMUET) and it [17]. It also uses a scene graph that controls the objects in virtual space. Panda3D implements several features, some of these use specific engines, such as, sound, physics, shading language and artificial intelligence. It provides support to effects, character animation, particle system, scripting language, input devices, graphic user interface, and network management, among others. In the next section, these 3D game engines are compared and analyzed.

4. Comparison and Analysis

The engines features were grouped into four groups (graphics, non-graphics, development and background) for organization purposes as described in the next sub-sections. For each group, the characteristics and technical issues are listed in tables, in alphabetical order. New features that go beyond the ones described in sources such as WebMaster.net, and are currently implemented by the engines, are described.

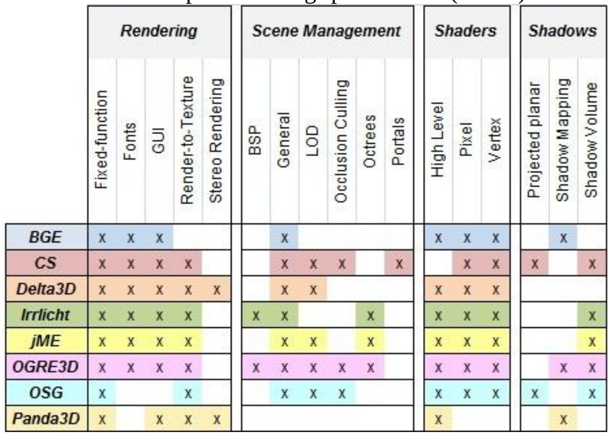
4.1 Graphics features

The graphics features include animation, curves and surfaces, lighting, meshes, rendering, texturing, scene management, shaders, shadows, special effects and terrain as shown in Table 1 (For organization purposes, Table 1 is partitioned in three parts). Animation, lighting, meshes, rendering, shaders, special effects and texturing are explored by all the engines at some level. Delta3D does not have support to shadows and Panda3D does not have scene management. Curves and surfaces are not implemented in Delta3D, Irrlicht, OSG and Panda3D.

Table 1: Comparison of the graphics features (Part I).

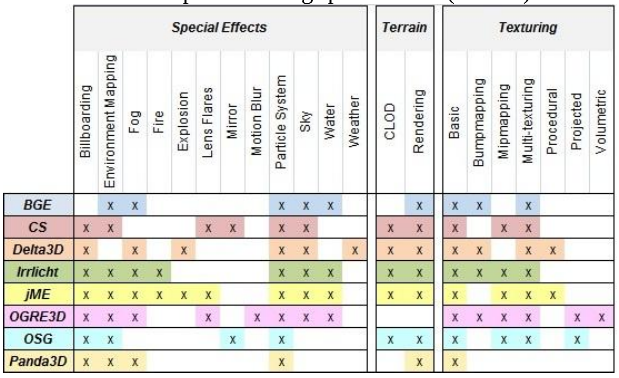
	Animation							Curves & Surfaces		Lighting				Meshes			
	Animation Blending	Facial Animation	Inverse Kinematics	Keyframe Animation	Morphing	Ragdoll	Skeletal Animation	Patches	Splines	Anisotropic	Lightmapping	Per-pixel	Per-vertex	Mesh Loading	Progressive	Skinning	Tessellation
BGE			x	x					x			x	x	x		x	
CS			x	x		x	x	x	x		x		x	x	x		
Delta3D							x					x	x	x			x
Irrlicht	x				x		x				x	x	x	x			
jME				x			x		x		x		x	x		x	
OGRE3D	x	x	x				x		x		x	x	x	x	x	x	
OSG	x			x			x			x			x	x		x	
Panda3D	x				x		x						x	x		x	

Table 2: Comparison of the graphics features (Part II).



	Rendering					Scene Management						Shaders			Shadows		
	Fixed-function	Fonts	GUI	Render-to-Texture	Stereo Rendering	BSP	General	LOD	Occlusion Culling	Octrees	Portals	High Level	Pixel	Vertex	Projected planar	Shadow Mapping	Shadow Volume
BGE	x	x	x				x					x	x	x		x	
CS	x	x	x	x			x	x	x		x		x	x	x		x
Delta3D	x	x	x	x	x		x	x				x	x	x			
Irrlicht	x	x	x	x		x	x			x		x	x	x			x
jME	x	x	x	x			x	x		x		x	x	x			x
OGRE3D	x	x	x	x		x	x	x	x	x		x	x	x		x	x
OSG	x			x			x	x	x			x	x	x	x		x
Panda3D	x		x	x	x							x				x	

Table 3: Comparison of the graphics features (Part III).



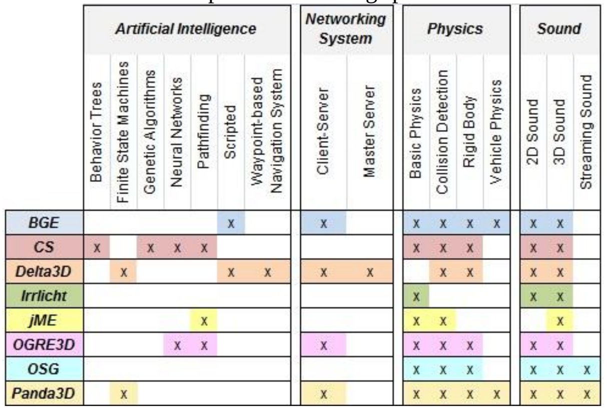
	Special Effects												Terrain		Texturing						
	Billboarding	Environment Mapping	Fog	Fire	Explosion	Lens Flares	Mirror	Motion Blur	Particle System	Sky	Water	Weather	CLOD	Rendering	Basic	Bumpmapping	Mipmapping	Multi-texturing	Procedural	Projected	Volumetric
BGE		x	x						x	x	x			x	x	x		x			
CS	x	x				x	x		x	x			x	x	x		x	x			
Delta3D	x		x		x				x	x		x	x	x	x	x		x	x		
Irrlicht	x	x	x	x					x	x	x		x	x	x	x	x	x			
jME	x	x	x	x	x	x			x	x	x		x	x	x		x	x	x		
OGRE3D	x	x	x			x		x	x	x	x				x	x	x	x		x	x
OSG	x	x					x		x				x	x	x		x	x		x	
Panda3D	x	x	x						x					x	x						

BGE now has support for the GLSL high-level shading language; CS supports the animation for ragdoll and inverse kinematics and the management of Level Of Details (LOD); Ogre3D has support for facial animation; and Panda3D has support for blending and morphing animation, billboarding, shadow mapping, render-to-texture and environment mapping.

4.2 Non-graphics features

The non-graphics features include artificial intelligence, networking, physics, sound and video.

Table 4: Comparison of the non-graphics features.



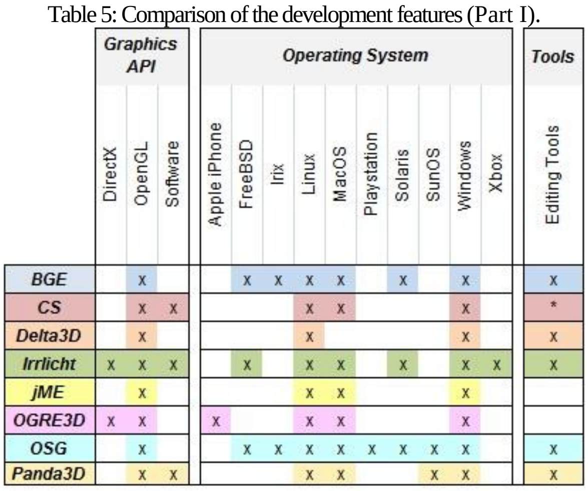
	Artificial Intelligence							Networking System		Physics				Sound		
	Behavior Trees	Finite State Machines	Genetic Algorithms	Neural Networks	Pathfinding	Scripted	Waypoint-based Navigation System	Client-Server	Master Server	Basic Physics	Collision Detection	Rigid Body	Vehicle Physics	2D Sound	3D Sound	Streaming Sound
BGE						x		x		x	x	x	x	x	x	
CS	x		x	x	x					x	x	x		x	x	
Delta3D		x				x	x	x	x		x	x		x	x	
Irrlicht										x				x	x	
jME					x					x	x				x	
OGRE3D				x	x			x		x	x	x		x	x	
OSG										x	x	x		x	x	x
Panda3D		x						x		x	x	x	x	x	x	x

All engines have some support to physics and sound. BGE, Delta3D, Ogre3D and Panda3D have support to client-server network. Only Delta 3D has support to networking system with HLA protocol. Artificial intelligence techniques are not fully explored by most engines. The last non-graphics features updating are as follows: CS has support for AI: behavior trees, genetic algorithms, neural networks and pathfinding [9]; Delta3D has support the client/server network and AI (finite state machine, waypoint-based navigation system and high-level scripting language) [18]; jME supports AI (pathfinding); and Ogre3D provides wrappers for AI (pathfinding), client/server network, and sound.

4.3 Development features

Development features include graphics APIs, operating systems, editing tools, programming languages, bindings, wrappers and scripting languages.

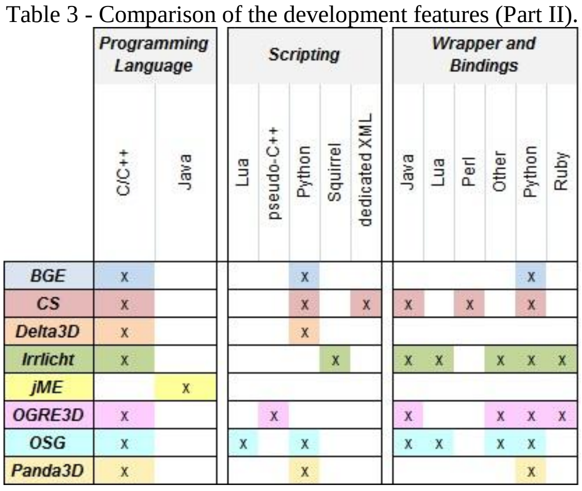
Table 5: Comparison of the development features (Part I).



	Graphics API			Operating System										Tools
	DirectX	OpenGL	Software	Apple iPhone	FreeBSD	Irix	Linux	MacOS	Playstation	Solaris	SunOS	Windows	Xbox	Editing Tools
BGE		x			x	x	x	x		x		x		x
CS		x	x				x	x				x		*
Delta3D		x					x					x		x
Irrlicht	x	x	x		x		x	x		x		x	x	x
jME		x					x	x				x		
OGRE3D	x	x		x			x	x				x		
OSG		x			x	x	x	x	x	x	x	x		x
Panda3D		x	x				x	x			x	x		x

* CS can be used in conjunction with the Blender editor.

Table 3 - Comparison of the development features (Part II).



	Programming Language		Scripting					Wrapper and Bindings					
	C/C++	Java	Lua	pseudo-C++	Python	Squirrel	dedicated XML	Java	Lua	Perl	Other	Python	Ruby
BGE	x				x							x	
CS	x				x		x	x		x		x	
Delta3D	x				x								
Irrlicht	x					x		x	x		x	x	x
jME		x											
OGRE3D	x			x				x			x	x	x
OSG	x		x		x			x	x		x	x	
Panda3D	x				x							x	

Since all engines have support to both Windows and Linux OS, they all use at least OpenGL graphics API. Only Irrlicht and Ogre3D include support to DirectX. Only Ogre3D has support to Apple iPhone, OSG to Playstation and Irrlicht to Xbox. Only Delta 3d does not have support to MacOS. The unstable version jME 3.0 has support to Android. The C/C++ programming languages are default, except for jME that was developed to Java. Python and Java are the main languages with wrappers or bindings support. Python is the main language for scripting. CS now provides bindings for Python, Java and Perl languages and also has support for the scripting language: Python or a dedicated XML compliant description; and Irrlicht provides binding for the Java, PureBasic, BlitzBasic, Lua, Ruby, Python and GameMaker. BGE, CS, Delta3D, Irrlicht and OSG have stable editing tools. Blender is a 3D editor that can be used with BGE or Crystal Space. The STAGE is the simulation, training, and game editor created by the MOVES Institute at the Delta3D. irrEdit is a 3D world as well as level editor for the Irrlicht engine. osgEdit is a scene graph editor for OSG. MonkeyWorld3D is a world editor for the deprecated jME 1.0 (the unstable jME 3.0 has a game development environment – jMonkeyPlataform).

4.4 Background features

The background features include the engine creation year, which can indicate the maturity of the engine; if there is documentation; the number of persons in the core team and the number of forum members (in thousands), which can indicate the community involvement level; license name defines the user's freedom; the update date of the current stable version, the current stable version and the latest unstable version can indicate if there is continued development. All engines have more than six years of development and had one stable version updated last year. Ogre3D license is MIT from 1.7 versions. BGE can be used commercially allowing access to source code, as well as CS, Delta3D and OSG. Irrlicht, jME, Ogre3D and Panda3D do not require distribution of source code for development of commercial games.

Table 6: Comparison of the background features.

	Engine			Forum	License Name						Version		
	Creation year	Documentation	Number of persons in core the team	Number of forum members	BSD	GPL	LGPL	MIT	new BSD	zlib/ libpng	Update date of current stable version	Current stable version	latest unstable version
BGE	1995	x	Blender Foundation	30-35		x					02/09/2009	2.49	2.53
CS	1997	x	4	1-5			x				28/11/2009	1.4.0	1.9.0
Delta3D	2004	x	MOVES Institute	1-5			x				06/11/2009	2.4.0	2.5.0
Irrlicht	2003	x	1	10-15						x	17/02/2010	1.7.1	na
jME	2003	x	5	5-10					x		09/09/2009	2.0.1	3.0
OGRE3D	2000	x	7	20-25				x			02/05/2010	1.7.1	2.0
OSG	1998	x	75	1-5			x				05/04/2010	2.8.3	na
Panda3D	2001	x	Disney Imagineering and CMUET Center	1-5	x						14/05/2009	1.6.2	1.7.0

4.5 Discussion

The tables in the previous sections can be used to facilitate the developer´s choice by providing a compact but general view. There is no better engine, but one that fits the needs of a specific project. This conclusion can be reached with a single feature or by combining more than one, as shown in the following few examples: **Project needs to be compatible with the network protocol HLA:** Delta3D is the engine that already provides this support (Table 2); **Platform restrictions:** (Table 3 - Part I) Xbox: Irrlicht; PlayStation: OSG; and iPhone: Ogre3D; **Project requires the native language Java:** jME is the only option (Table 3 - Part II); **Game in client-server network with non-player characters (NPCs) and collision detection:** In this case, it can be combined the support for client-server network with pathfinding (AI) for NPCs, collision detection (physics) and facial animation. As can be seen in Table 1 (Part I) and Table 2, Ogre3D is the engine that covers all these features; and **Standalone racing cars with special effects to simulate the sky:** In this case, vehicle physics and sky (special effects) can be combined, with the use of curves to control the path of other vehicles (NPCs). As can be seen in Table 1 (Part I and III) and Table 2, BGE meets all these needs.

5. Conclusion

This paper presented the best open source 3D game engines with support to Windows and Linux Operating Systems. A compact view was presented of all main features grouped as graphics, non-graphics, development and background features. Other important aspects were mentioned, such as, maturity, community size and activity level. Each game project has specific needs. Game engines are of great help to games developers as they provide an abstraction that hides complex features from the user and the application. This paper contributes to a more general understanding of the eight best considered game engines available for game and simulation developers.

Acknowledgements

The authors wish to thank CAPES, CNPq and FAPESP for the support to INCT-SEC Project (proc. 573963/2008-8 and 08/57870-9.

References

[1] MARKS, S., WINDSOR J. AND WÜNSCHE, B. 2007. "Evaluation of Game Engines for Simulated Surgical Training". *In: Proc. 5th Intl Conf. on Computer graphics and interactive techniques in Australia and Southeast Asia*, 01-04 December 2007 Perth. Australia: ACM, 273-280.

[2] FEIJÓ, B. ET AL. 2010. Introdução à CC com Jogos: aprendendo a programar com entretenimento, RJ: Elsevier.

[3] EVES, G., AND MEEHAN, P. 2008. The development of selection criteria for game engines in the development of simulation training systems. In www.siaa.asn.au/get/2451303127.pdf

[4] ELIAS-PROJECT. 2005. *Open Source & Low Cost Game Engines*. http://ludocraft.oulu.fi/elias/dokumentit/open_ source_ game_engines.pdf

[5] LACORT, A. M. 2009. Integration of a 3D rendering engine with a physics simulator. Master thesis, Lleida University.

[6] http://www.dev master.net/engines/search.php

[7] http://www.blender.org/

[8] http://www.crystalspace3d.org/main/Main_Page

[9] TYBERGHEIN, J. ET AL. 2010. Developing video games and VEs with the Crystal Space engine, *DEnG-VE workshop, EICS conference*.

[10] MCDOWELL, P. ET AL. 2006. Delta3D: a complete open source game and simulation engine for building military training systems. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 3, 143-154.

[11] http://www.delta3d.org/

[12] 1516-2000. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules, *IEEE Computer Society*.

[13] http://irrlicht.sourceforge.net/

[14] http://www.jmonkey engine.com/home/

[15] http://www.ogre3d.org/

[16] http://www.openscenegraph.org/projects/osg

[17] http://www.panda3d.net

[18] DARKEN C.J. ET AL, 2007. Game AI in Delta3D. *In: Proc of the 2007 IEEE Symposium on Computational Intelligence and Games (CIG 2007).*