Adaptive Physics for Game-balancing in Videogames

Carlos Delgado-Mata[1], Jesús Ibáñez[2], and Alfredo Villalobos[1]

[1] IPIT (Instituto Panamericano de Investigación Tecnológica),
Escuela de Ingenieria, Universidad Panamericana campus Bonaterra,
Aguascalientes, México, C.P. 20290 cdelgado@up.edu.mx
[2] Grupo de Tecnologias Interactivas,
Departmento de Información y Tecnologias de Comunicación,
Universitat Pompeu Fabra,
Barcelona, España

Abstract. This article describes the work in progress of an adaptive video game that is based on the arcade game of air hockey. The video game uses adaptive physics to improve the experience between two human opponents. The physics of the game continually adapts to the ability of each player. That is, the game becomes more difficult for the skilful player; however, it becomes easier for the inexperienced player. This is achieved by using a behaviour based architecture; first used on mobile robots and later on artificial characters and computer driven video game opponents. The architecture affects the physical properties of some elements of the game, which are handled by the physics engine used in the project described herein.

Keywords: adaptive physics, physics engines, game balancing

1 Introduction

Video games have reached a near photo-realism graphical level. Therefore, virtual images can be confused with the images of reality. This creates a high expectation on video game players; however, the performance of some computer driven opponents does not correspond to the level of graphical realism. Thus, the user's perception is drastically affected. To resolve this problem, research has been carried-out to explore various techniques to improve the believability of the characters' behavior. In [8] a discussion on characters' believability in virtual environments is presented. Another example is that of the game Überpong [9], where the the behaviour of opponents are provided with a personality profile to make them more believable.

1.1 Adaptive Games

To improve the user's experience, research has also been carried out on video games that adapt to the players' ability, and thence to enrich their gaming experience. One such example is the so-called *game balancing*. That is, the game continually changes so that the player enjoys the gaming experience (and therefore plays longer). Moreover, he should not get frustrated if the game is too complicated and he should not get

© *C. Delgado, C. Gutiérrez, R. Velazquez, H. Sossa (Eds.)* *Received 23/01/11*
Advances in Computer Science and Electronic Systems *Accepted 20/02/11*
Research in Computing Science 52, 2011, pp. 37-46 *Final version 07/03/11*

bored if the game is too simple. An example of this research is to modify the complexity in case of one player [12]. Similarly, there is research on games that are modified (adapted) depending on the player's emotional state. A relevant example is presented in [20], where one game mechanic (opponent speed) is continuously adjusted depending on the player's emotional state. The data was used to predict the mode of play (boring, frustrating or enjoyable), and depending on it the game changes the speed of the opponent accordingly. Furthermore, preliminary results have been published on research on the modification of complexity in a game between two human opponents [15]; the objective of game is social rather than competitive. This type of game is appropriate where a player is experienced (for example a parent) and the other is novice (for example a preschooler). The work described herein uses adaptive physics to enhance the game experience for two human opponents.

1.2 Physics in video games.

As mentioned above, the level of graphical realism in the latest video-game generation is close to photo-realism. Because of this, the industrial and the academic community have decided to improve the objects' behaviour. Thus, an steep increase in the use of physics engines in video games has occurred. Moreover, some video games are predominantly a physical simulation, some standout examples are: The game produced by renowned filmmaker Steven Spielberg, Boom Blox, where objects interact according to their physical properties, such as mass, acceleration, deformation and gravity. Another game that uses a physics engine predominantly to manage the behaviour of the game's elements is The World of Goo; where the player creates physics based constructions (for example bridges) to move balls (Goo) from one place to another. Other games have been used as benchmarks for physics engines; one such example is Cellfactor, which was developed by the colombian company Immersion Games. This game was used as a demonstration of power of the Ageia PhysX chip. To add physics to video games it is best to use physics engines; some of them are commercial and some are open source.

The relevant commercial physics engines are:

1. Havok, an irish library [11], is used in hundreds of games developed for the console market leaders (Wii, PS3 and Xbox 360), as well as, games developed for the PC. Some of the standout games developed with Havok are: The excellent adventure game developed for PS3 and Xbox 360: Assassins Creed, and the games Super Smash Bros. Brawl and Boom Blox developed for the Wii.
2. PhysX is a physics library first acquired by Ageia and subsequently by Nvidia (the market leader in graphics processing units GPUs) [17]. PhysX is now included as a physics process unit (PPU) in an integrated circuit. PhysX is used in hundreds of video games, most notably are the acclaimed Batman: Arkam Asylum and the successful Unreal Tournament 3.

The relevant open-source physics engines are:

1. Box2D is a 2D physics library [4] developed by Erin Catto. Box2D's main advantage is that it is multi-platform. The library has been used in various languages and environments such as C++, Java, Flash, iPhone and Android.

Fig. 1. Air hockey table.

2. Bullet physics engine is a 3D engine [5], which was developed by Erwin Coumans. The engine has been used in video games and in movie effects. It is available for several platforms, such as the PlayStation 3, Xbox 360, Wii, Mac, iPhone, Linux and Windows. It has been used in various games and movies; notable examples are the successful video game Grand Theft Auto IV and the 2012 Hollywood film.
3. Chipmunk Physics Engine is a 2D physics engine [14] developed by Scott Lembcke. This library is similar to Box2D, but it is simpler. It is primarily used for iPhone applications.
4. Open Dynamics Engine is a 3D physics engine [18], which development began in 2001. It has been used in games like Bloodrayne 2.
5. Tokamak is an open source physics engine [13] developed by David Lam, no information was found on relevant video games using Tokamak.

2 Adaptive Physics

The video game described herein is based on the arcade game Air Hockey.

Air hockey is a game that consists of a table, two paddles and a puck (see Figure 1). The game is an abstraction of the popular canadian game of Hockey. The board consists of a flat surface surrounded by a railing to prevent the puck and the rackets from flying out of the table. The most recent table models include a device that provide an air-cushion to reduce friction. In the two smallest rail segments there are holes where the puck enters. That is, they act as goals.

The configurable items on the air hockey game (see Figure 2) proposed in our work are:

Fig. 2. Configurable items.

1. Player's 1 goal. The size of the goal of player 1 can be modified as follows:

$$goal1_{length}(t) = goal1_{length}(t-1) + \Delta goal1_{length} \tag{1}$$

   That is, depending on the player's skill, the goal's size can increase or decrease.
2. Player's 1 paddle. The mass / volume of a player's 1 paddle could be modified as follows:

$$r1_{mass}(t) = r1_{mass}(t-1) + \Delta r1_{mass} \tag{2}$$

$$r1_{vol}(t) = r1_{vol}(t-1) + \Delta r1_{vol} \tag{3}$$

   The density does not change, ie

$$\frac{r1_{mass}(t)}{r1_{vol}(t)} = \frac{r1_{mass}(t-1)}{r1_{vol}(t-1)} \tag{4}$$

3. Puck. Mass / volume of the "puck" could be modified as follows:

$$p_{mass}(t) = p_{mass}(t-1) + \Delta p_{mass} \tag{5}$$

$$p_{vol}(t) = p_{vol}(t-1) + \Delta p_{vol} \tag{6}$$

   The density varies, ie

$$\frac{p_{mass}(t)}{p_{vol}(t)} \neq \frac{p_{mass}(t-1)}{p_{vol}(t-1)} \tag{7}$$

4. Player's 2 paddle. The mass / volume of the player's 2 paddle could be modified as follows:

$$r2_{mass}(t) = r2_{mass}(t-1) + \Delta r2_{mass} \tag{8}$$

$$r2_{vol}(t) = r2_{vol}(t-1) + \Delta r2_{vol} \tag{9}$$

   The density does not change, ie

$$\frac{r2_{mass}(t)}{r2_{vol}(t)} = \frac{r2_{mass}(t-1)}{r2_{vol}(t-1)} \tag{10}$$

5. Player's 2 goal. The size of the goal of player 1 could be modified as follows:

$$goal2_{length}(t) = goal2_{length}(t-1) + \Delta goal2_{length} \tag{11}$$

   That is, depending on the skill of the player's goal can be made larger or smaller.
6. Table's surface. The surface can be modified to make it rougher or smoother by affecting the air cushion which is provided by air hockey tables. The abstraction of this change is the modification of the friction constant.

$$\mu(t) = \mu(t-1) + \Delta\mu \tag{12}$$

7. Table configuration. That is, alter the length and width of the table.

$$table_{length}(t) = table_{length}(t-1) + \Delta table_{length} \tag{13}$$

$$table_{width}(t) = table_{width}(t-1) + \Delta table_{width} \tag{14}$$

Of the seven aforementioned configurable items, the table's surface and the puck's physical properties were selected. This decision was based on the feasibility to replicate the changes on a real air hockey table (like the one shown in figure 1).

The elements of a real table that can feasible be modified are:

- Table's surface. The friction between the table and the "puck" can be altered by changing the amount of air provided by the air pump.
- Player's 1 goal. The goal of player 1 can be modified with a device similar to that found in automatic sliding doors. That is, the goal can change it's length.
- Player's 2 goal. The goal of player 2 can also be modified with a device similar to that found in automatic sliding doors. That is, the goal can change it's length.

It is easier to change the size of the goals in a real table than it is to modify the properties of a real puck; never the less, the end result is similar: it make it easier or harder to score a goal.

3 Adaptive physics for video games

The ultimate goal of the work presented herein is that both players enjoy the game, regardless of the level of each player. That is, the game must be neither too complicated for the inexperienced player, because the player could get frustrated; or too simple for the skilled player, as the player could get bored.

To achieve this, the strategy described in [15] was used, where the following objectives have been pursued:

- The number of points per player must be balanced. That is, the performance of both players should be as close as possible so that both remain engaged with the game.
- The number of times each player hits the puck should not be too small nor too big. Few hits would make the game frustrating; whereas many hits would make the game boring.

To achieve these objectives, a set of parameters were selected.

3.1 Parameters

The configurable items (mentioned in section 2)

- Physical properties of the puck.
- Table surface.

To achieve the aforementioned objectives the following control variables were assigned.

- Goal difference. If the goal difference is large it implies that there is a significant skill difference between the players. The following formula indicates the case that if the goal difference is greater than a threshold the physics of the game must be adapted to make the game more entertaining.

$$abs(p1_{goals} - p2_{goals}) > goals\_treshold \tag{15}$$

- Number of hits to the puck before scoring a goal. If the number of hits to the puck is very large it implies that the game is boring; whereas if the number of hits the puck is small it means that the game is frustrating; in either case the physics of the game must adapt to make the game more entertaining.

$$p1_{hits} + p2_{hits} > max\_hits\_threshold \tag{16}$$

$$p1_{hits} + p2_{hits} < min\_hits\_threshold \tag{17}$$

If either threshold is exceeded, the physics of the game, must adapt to make the game more entertaining.

3.2 Artificial intelligence for generating adaptive physics.

Research on behaviour based robotics has shown that a character can perform complex and interesting behaviour [8][9] with the use of simple rules.

This research field was inspired by the work of Valentino Braitenberg [2]. Some of the robots that he proposed could be interpreted as more complex than the simple rules used to implement them. For example, he named the behaviour of some his proposed robots as: *love*, *fear* and *aggression*. Similar work is the horizontal robotic architecture put forward by Brooks [3]. This architecture, in turn, was the inspiration behind the BSA (Behavioural Synthesis Architecture); first used in cooperative robots [1]. This architecture was expanded to communicate emotions through artificial pheromones on virtual mammalian species [6]. This architecture was thence enhanced to affect the behaviour on groups (flocking) through an emotion based architecture [7]. Recently, a derivate of the above architecture was developed to provide opponents with personality profiles: aggressive, fearful, pathetic, bold, cautious, impulsive, predictable or analytical [10].

In this paper, we propose the use of a behaviour-based architecture to provide a video game (air hockey) with adaptive physics and it is described next:

3.3 Adaptive physics for the air hockey game.

Adaptive physics for the air hockey game is defined by the two parameters shown in figure 3 and described next.

1. The coefficient of friction for the table's surface material. This simulates the air cushion provided in the real air hockey table, like the one shown in figure 1. This parameter is affected by the following behaviours: avoid boredom and avoid frustration. These behaviours receive the stimulus: number of puck hits before scoring a goal.
2. The physical properties of the puck like volume and mass mentioned in section 2. This parameter is affected by the following behaviours: assist player 1 and assist player 2. These behaviours receive the stimulus: score difference.

4 Development

The air hockey game with adaptive physics is being developed in the programming language C++, the OGRE graphics library [19]; whereas, the physics engine used in the project is Bullet [5]. The models have been modeled in Blender [16]. Figure 4 shows the process to export a blender modeled geometry to the OGRE graphics library.

Figure 5 shows the rendering cycle for the adaptive physics game. The steps in the rendering cycle are:

- Read the input device
- Perform the video game logic (in this case for an air hockey game).
- Animation of the elements of the game.

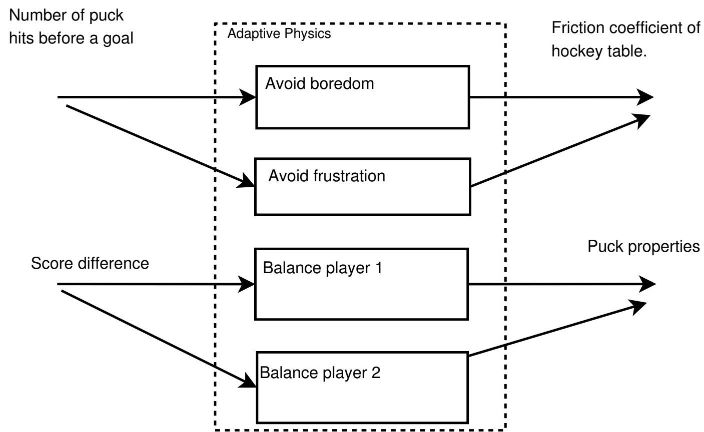

Fig. 3. Adaptive physics architecture.

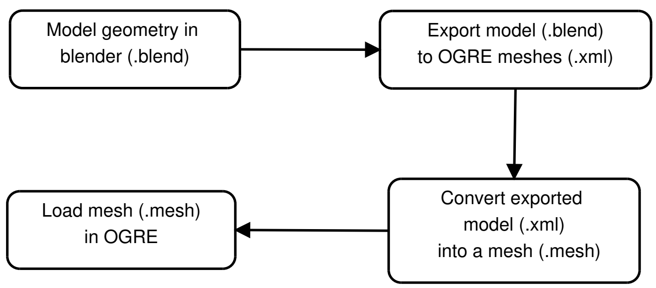

Fig. 4. Process from modeling in Blender to import a geometry in OGRE.

- Physics-based adaptive behaviours and described in the previous section.
- Run the physics engine (Bullet) simulation.
- Render the frame on the screen using OGRE.

The result is shown in figure 6.

5 Conclusions and Future Work.

The work in progress presented in this paper is an effort to improve the experience of players with different skill levels. This is accomplished by adjusting the physical parameters of some game elements, so that the game does not become boring for the skilled player or frustrating for the inexperienced player. For future work, experiments will be carried out to verify that the player's experience is significantly improved by adapting the game to the different player abilities. Furthermore, a real air hockey table will be developed and further experiments will be carried out therein. The results of both experiments will thence be cross-referenced.

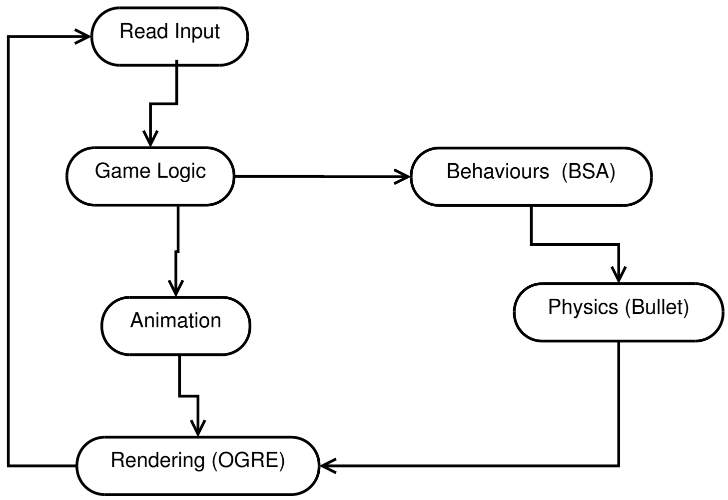

Fig. 5. Rendering cycle for the adaptive physics game.

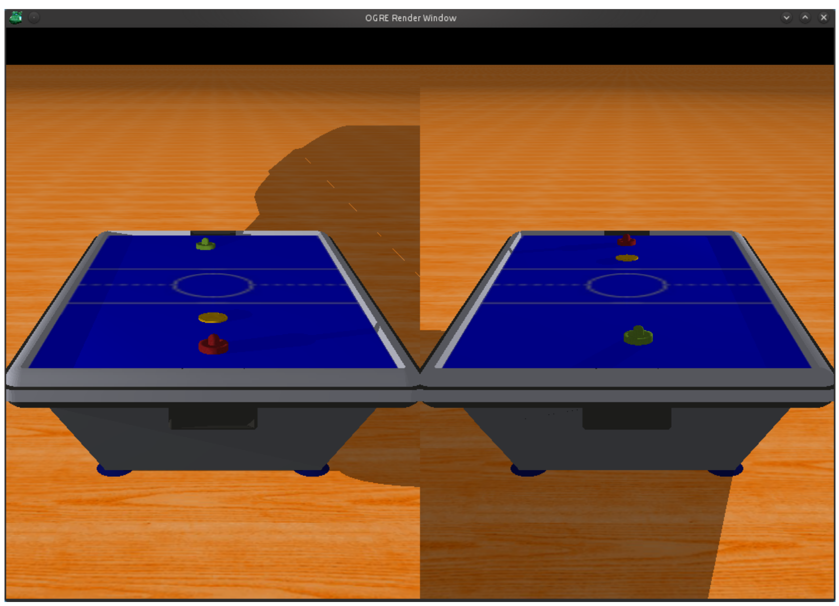

Fig. 6. Screen taken from air hockey game with adaptive physics.

References

1. Barnes, D.P.: A behaviour synthesis architecture for cooperanting mobile robot. In: Gray, J.O., Caldwell, D.G. (eds.) Advanced Robotics and Intelligent Machines, pp. 295 – 314. IEE Control Engineering Series 51, United Kingdom (1996)

2. Braitenberg, V.: Vehicles: Experiments in Synthetic Psychology. Papermac, Oxford, United Kingdom (1984)
3. Brooks, R.A.: A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation 2(1), 14–23 (1986)
4. Catto, E.: Box 2d, http://www.box2d.org/
5. Coumans, E.: Bullet physics library, http://bulletphysics.org/wordpress/
6. Delgado-Mata, C., Aylett, R.: Communicating emotion in virtual environments through artificial scents. In: de Antonio, E., Aylett, R. (eds.) LNCS Intelligent Virtual Agents. Springer-Verlag, Madrid, Spain (2001)
7. Delgado-Mata, C., Ibanez, J., Aylett, R.: Let's run for it: Conspecific emotional flocking triggered via virtual pheromones. In: Butz, A., Krüger, A., Oliver, P. (eds.) LNCS Smart Graphics 2003. pp. 131–140. Springer-Verlag, Heidelberg, Germany (2003)
8. Delgado-Mata, C., Ibáñez-Martínez, J., Bee, S., Ruiz-Rodarte, R., Aylett, R.: On the use of virtual animals with artificial fear in virtual environments. New Generation Computing 25(2), 145–169 (2007)
9. Delgado-Mata, C., Ibáñez-Martínez, J., Gómez-Caballero, F., Guillén-Hernández, O.M.: Behavioural reactive agents to define personality traits in the videogame überpong. T. Edutainment 1, 135–149 (2008)
10. Delgado-Mata, C., Ruvalcaba-Manzano, R., Quezada-Patino, O., Gomez-Pimentel, D., Ibanez-Martinez, J.: Low cost video game technology to measure and improve motor skills in children. In: IEEE Africon. pp. 1–6. Nairobi, Kenya (September 2009)
11. Havok: Havok physics, http://www.havok.com/
12. Hunicke, R.: The case for dynamic difficulty adjustment in games. In: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology. pp. 429–433. ACE '05, ACM, New York, NY, USA (2005), http://doi.acm.org/10.1145/1178477.1178573
13. Lam, D.: Tokamak physics engine, http://www.tokamakphysics.com/
14. Lembcke, S.: Chipmunk, http://code.google.com/p/chipmunk-physics/
15. Ibáñez Martínez, J., Delgado-Mata, C.: From competitive to social two-player videogames. In: Proceedings of the 2nd Workshop on Child, Computer and Interaction. pp. 18:1–18:5. WOCCI '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1640377.1640395
16. Mullen, T.: Mastering Blender. Sybex, USA (2009)
17. Nvidia: Nvidia physx, http://developer.nvidia.com/object/physx.html
18. Smith, R.: Open dynamics engine, http://www.ode.org/
19. Streeting, S.: Ogre: Open source 3d graphics engine, http://www.ogre3d.org/
20. Tijs, T., Brokken, D., Ijsselsteijn, W.: Creating an emotionally adaptive game. In: Proceedings of the 7th International Conference on Entertainment Computing. pp. 122–133. ICEC '08, Springer-Verlag, Berlin, Heidelberg (2009), http://dx.doi.org/10.1007/978-3-540-89222-9_14