Integration of a Physics Engine into an Adventure Game Platform

Ángel Serrano, Eugenio J. Marchiori, Ángel del Blanco, Iván Martínez-Ortiz, Baltasar Fernández-Manjón

***Title*— Integration of a physics engine into an adventure game platform.**

***Abstract*— This paper presents the challenges and possibilities that arise from the integration of a 2D physics engine into an adventure game platform (i.e. engine and editor). This integration allows the use of a complex physics model within a narrative environment, thus increasing expressiveness and educational potential (through re-usability and increased challenge). In this work eAdventure is used as the adventure game platform and Box2D is used as the physics engine. While adventure games have been successfully applied in educational contexts from language learning to medicine, including a physics engine broadens the number of subjects that can be covered, thus increasing the scope and potential of the resulting games. This paper shows how such integration was implemented, and shows several mini-games that use the physics engine to provide visualization of physical experiments and attempt to increase player engagement. The mini-game approach allows for the easy integration of these new physics-based interactive scenarios within an adventure game narrative, where each mini-game can represent a puzzle or challenge to be solved.**

***Index Terms*—Computer aided instruction, Computer simulation, Physics education, Visualization**

I. Introduction

THE use of educational video games is growing ever more popular in the past few years. Recent studies show that games can increase student engagement during instruction [1], [2], academic achievements [3], [4] and some skills, knowledge and attitudes [5-7]. Other studies show that investment in this area is increasing [8], which shows growing interest not only in research but by private companies as well.

Simulation is one of the most powerful tools for learning in computer-based environments. Simulations can be used to visualize processes, to provide context and to allow users to experience events as many times as they want in controlled environments where meaningful feedback is provided. Simulation and game-based simulations have been used in successful learning experiences in fields from medicine [9] to business [10]. Simulations often prove an alternative to other approaches were cost is a concern, or where ethical limits on the repetition of experiments for educational purposes apply [11].

Simulations can be more effective when integrated in broader environments that provide a richer educational context (e.g. intelligent tutoring systems). We consider that the combination of simulations and games is very promising because it allows the extension of the game metaphor to create better educational games (or game-like simulations) by including accurate physics models and, at the same time, keeping the main game advantages (e.g. engagement, feedback).

In the physics field, in particular in Newtonian mechanics, authors like Forbus [12] have pointed out that it is easier to learn physics through visual experiments than mathematical formulas. To simulate visual experiments, there are different approaches, from using 2D or 3D *physics engines* that allow the integration of physical environment simulations to programming the simulations from scratch using different tools (e.g. *Adobe Flash*, *Microsoft Excel*, etc.) or programming languages (e.g. *Java*). Unlike using non-specific tools where the mathematical model has to be implemented, *physics engines* simplify the development because they already include a mathematical model of the physical aspects (e.g. force, movement) that is applied in a virtual environment, having similar effects as those experienced in the real world. Moreover, using physics engines increases code maintainability and allows for the validity of the model to be independently assessed.

Some authors have explored the use of a physic engine in learning processes [13], with some positive results. Examples of visual experiments can be found in the web (usually created using *Adobe Flash*), showing physical simulations of different kinds. But most of them are only intended to show the simulation process, with limited or no user interaction and with very limited feedback. Applications with user interaction and complex physics models are available in many popular games, where the physical simulation is used to provide engagement to players (e.g. *Angry Birds*, a game downloaded over 400 million times[1]).

Á. Serrano, E. J. Marchiori, Á. Del Blanco, I. Martínez-Ortiz & B. Fernández-Manjón are with the School of Computer Science, Complutense University at Madrid, 28040 Madrid, Spain (corresponding author e-mail: aserrano@e-ucm.es).

DOI (Digital Object Identifier) Pending

[1] http://techcrunch.com/2011/10/18/after-400-million-downloads-angry-birds-introduces-new-bird-movie-confirmed/

ISBN: 978-84-8158-561-2

We propose the use of the physic engine as part of the gameplay, built within the newest eAdventure platform [14]. eAdventure is an educational game platform, allowing for both the creation and deployment of games. eAdventure games are traditionally of the 2D *point-and-click* genre, where the player follows a story (i.e. adventure games). The new approach we propose is to extend the previous game metaphor introducing new physics simulations and visualizations as mini-games, integrated with the story, which the player has to go through to advance in the story. The mini-games will represent puzzles or challenges that the player must solve to move forward in the story.

The integration of these physics mini-games is done both in the engine (where the physics engine is used for visualization and interaction with the simulation) and in the editor, where mini-games can be defined and included in complex narrative environments. Moreover, the physics engine can be used in other parts of the platform, to complement the representation of environments by increasing the number of interactive elements that the player can, for instance, knock over.

This paper is structured as follows: First, we address the technological problems derived from the integration of a physics engine with a game engine. Second, we introduce how physics simulation can be configured through the game editor. Then, we present examples of mini-games based on physics, and finally provide some final remarks and future work.

II. Technical Approach

In our approach we propose the integration of an existing physics engine into a pre-existing educational game platform. The eAdventure 2.0 educational game platform is used [15], which traditionally allows for the creation of educational adventure *point-and-click* games. The game engine in the eAdventure platform (i.e. the eAdventure engine) uses a model that must be adapted for the use of physics simulations, as required by the physics engine.

There are several popular 2D open code physics engines available that can be integrated into existing systems, such as Box2D[2], Chipmunk[3] or Fraser[4]. A common characteristic of all these engines is that they use entities called *bodies* as minimal simulation units. Using bodies all the objects and parameters required to represent, initialize and perform the physic simulation (Fig. 1) are defined. Typically, this information includes:

- Collision domain, defined from polygon shapes (usually rectangles and circles), which also establishes the shape of the object (used during collision detection)
- Coefficient of restitution, a value representing the ratio of speeds after and before an impact with another object (used to allow for bouncing)
- Coefficient of friction, a value representing the ration of the force of friction and the force pulling two objects together (used when there is contact between two objects)
- Mass of the body, which can be proportional to the area of the collision domain or can be established for the different elements in a simulation
- The position of the body in the world, defined by the x and y coordinate of the body center
- The rotation of the body about the horizontal x-axis line

Moreover, there are two main body types: statics bodies (i.e. walls, obstacles and other elements that don't change their position and rotation during the simulation, but can collide with other bodies), and dynamics bodies (e.g. moving bodies that can be affected by forces and impulses, changing their rotation and position during simulation, and that can collide with other statics or dynamics bodies).

Given that *Box2D* is release under the LGPL license, is used in many games and simulations, and has a Java and GWT (*Google Web Toolkit*) implementation, we consider it the most appropriate for the multi-platform goals of the eAdventure 2.0 platform [15]. *Box2D* is also the same physics engine used in the successful and multi-platform *Angry Birds* games, which shows its potential for real-world use scenarios. However, the approach can be easily modified to apply any other 2D physics engine.

In the eAdventure game engine, the minimal unit is the *GameObject*. A *GameObject* is the basic element in the 2D game environment, which can be as complex as the whole game scene (everything the player can see) to a single element that appears in a game scene. It has a graphical representation (an image, a shape or a text), position (x and y coordinates), rotation, scale, width and height, and other additional parameters. Besides, additional customized values can be assigned to *GameObjects*. Some of these data will be used to build the required physical bodies for the simulation.

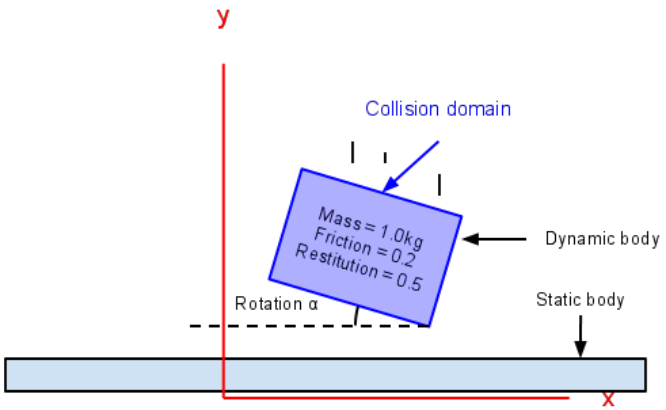

Fig. 1. Graphical representation for bodies in most physics engines. The main data is position and rotation, and whether the body is static or dynamic.

[2] http://box2d.org/
[3] http://code.google.com/p/chipmunk-physics/
[4] http://farseerphysics.codeplex.com/

ISBN: 978-84-8158-561-2

The physics simulation process requires the eAdventure *GameObjects* to be converted into *Box2D* bodies, and in each game cycle (i.e. the lapse of time between model changes) the physics simulation must be updated.

In eAdventure, the game logic is represented through *effects*. For example, when a game event occurs (e.g. a mouse interaction, a timed event), an *effect* can be triggered (e.g. show a text, change a variable's value). In our approach with Box2D, we used an *effect* to launch and update the simulation (Fig. 2). This way the impact of the inclusion of the physics engine is limited to one element in the engine (i.e. the *PhysicsEffect*), simplifying maintenance and the replacement of the engine for another one.

In the data of the *effect*, we add all the *GameObjects* that will take part in the physic simulation, with some additional data that might not be possible to extrapolate, such as mass, coefficient of restitution, coefficient of friction or whether the body is static or dynamic. The other parameters required by the body (collision domain, position and rotation) are created from the *GameObject* data.

When the *PhysicsEffect* is launched, *GameObjects* are translated into physic bodies. Then, the simulation is started. To keep track of the simulation, all the *GameObjects* are linked to their physical body through a variable. When the *effect* updates, the simulation takes a step and then updates the *GameObject* position and rotation from the physical body linked to it.

The physical simulation will be updated 15 times per second, as is the main loop of the eAdventure game. The phyics engine has different mechanisms (e.g. continuos collision detection) to make sure that collisions and other events take place as expected. Moreover, animations can be improved by the use of an interpolation factor that, together with the speed of the different elements in the scene, can be used to recreate intermediate frames and improve the smoothness of the animation.

III. Physic Game Authoring

The introduction of a physic engine greatly extends the representational power and interactions that are possible in the games, but this flexibility also increases the game authoring complexity.

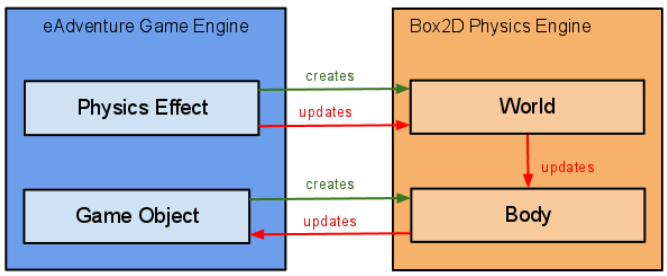

Fig. 2. eAdventure game engine elements relation with the Box2D engine. The eAdventure effect creates a body for each *GameObject* and updates the Box2D physical model. Changes in the body, performed by the world update, are reflected back in the *GameObject*.

To address this complexity issue in the physic game authoring we implement two different approaches: a) a simplified approach, where only a few parameters of a pre-set mini-games can be edited; and b) a more advanced and complete approach, where the entire physics scene and all its elements can be precisely defined. As the approaches affect the amount of information available, this has a direct relationship with the difficulty of the game development. In most scenarios, the game authoring approach using the simplified approach should be adequate for teachers, including enough configurable options to cover most problem types.

In the simplified approach, a prebuilt mini-game is offered. In the next section, we present three examples of three prebuilt mini-games. For each of these mini-games, a limited number of parameters can be edited (e.g. the scene background, units of measure, the unknown parameter) (Fig. 3). Moreover, these games can be integrated in the main story in different ways, depending on how the outcome of the game is used. The teacher can set the limit rounds the mini-game can be played and the following scene in the game if the mini-game was completed successfully or not. Moreover, effects can be associated to these scene changes, which can be used, for instance, to increment the global score of the player in the game.

In the advanced approach, an entire physic scene can be created from scratch. The advanced editor, also used in the authoring of other games parts, allows for the addition of new elements to the scene.

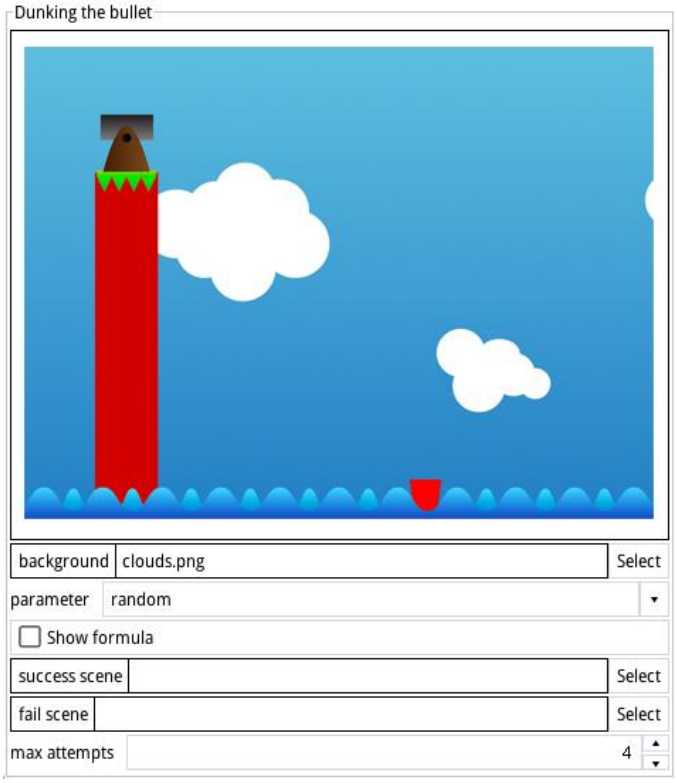

Fig. 3. Simplified approach for one of the mini-games presented in section IV. Several parameters can be edited for this mini-game, like the background or the parameter the student must fill.

Martín Llamas Nistal, Inmaculada Plaza García, Carlos Vaz de Carvalho, editores
ISBN: 978-84-8158-561-2

Each element in the scene can be assigned physical attributes (e.g. mass, friction, etc.), as mentioned in the previous section. This complex editor provides great flexibility to the authors, but is only useful and intended for advanced users that have an adequate background and knowledge about the physical modeling and game authoring in eAdventure.

For most cases, in this advanced edition, it would be necessary to add additional eAdventure elements (e.g. GUI interactions, conditions, operations and effects) to create full functional games.

IV. MINI-GAMES

As in other approaches, the basic use for a physic simulation is the visualization of processes that can be too abstract or too complex to be understood just with the mathematical expression. This visualization can be achieved with simpler non-interactive media such as a video, or a pre-programmed animation, as it does not require changes to the model in any way.

Giving the student the ability to modify one or more of the variables that affect the simulation can improve this visualization making it interactive. To achieve this, a physics model is required, as the simulation will vary depending on the value the student gives to those variables. For example, the student can be allowed to directly manipulate the length of the string in a pendulum and see how the period changes (Fig. 4). This limited intervention in the simulation cannot be considered a full mini-game as the interaction is not related with a game goal. However, this simple interactive visualization has enough educational value to be considered useful in some cases (e.g. when the teacher wants to explain the physic law just before to introduce a mini-game that uses this concept).

Another way to complement the visualization is to use it to prove or disprove the accomplishment of a goal. For instance, the previous example can be improved by making it a goal that the pendulum achieves a pre-set period by changing the length of the string. This simple change takes advantage of the simulation to show how the change affects the period, and to allow the player to test preconceived theories and concepts about how this works.

We present three more examples where the player needs to apply physics formulas and mathematics to calculate the expected results.

A. *Dunking the Bullet*

The challenge in "Dunking the bullet" is to get the bullet fired by a canyon into a floating basket on the water. The game is based on trajectories theory, particularly range of projectile theory. Result of the game is dependent on four parameters: the canyon height from the sea (h), the horizontal distance between the canyon platform and the basket (x), the angle between the canyon and the horizon (β) and the initial velocity for the bullet when the canyon fires it (v_i) (Fig. 5). In order to achieve this mini-game's goal the player will need to correctly apply equations of motion.

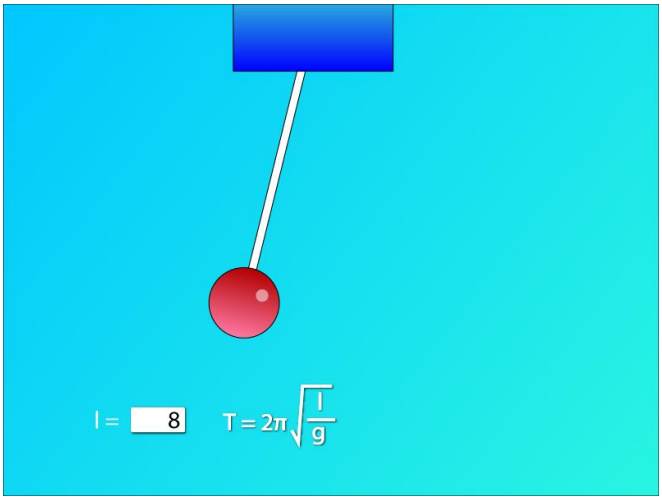

Fig. 4. Visualization of the period of oscillation of a pendulum. The length of the pendulum can be changed to see the simulation of the difference.

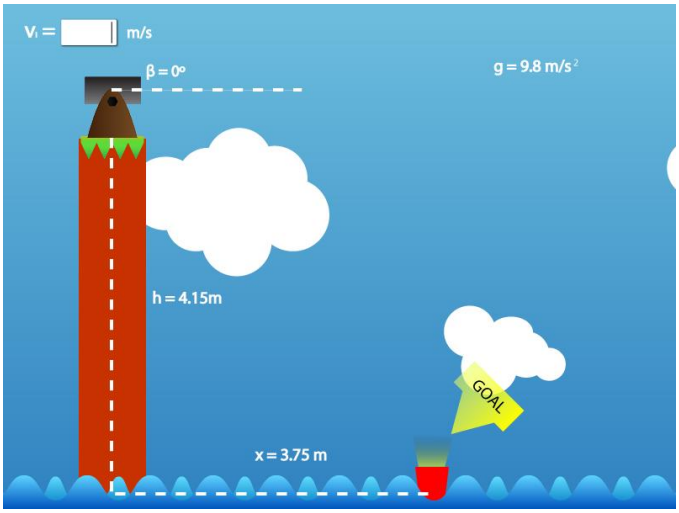

Fig. 5. In this phase of "Dunking the bullet", the unknown parameter that needs to be provided by the student is the initial velocity for the bullet. The other three parameters have random values.

The game is based on phases. When a phase starts, one parameter is unknown, and the other three are given random values. With these values, the player must calculate the unknown parameter and enters it into a textbox in the game. When a value is provided, the game starts the physical simulation, and if the calculations were right, the bullet enters in the basket, and the student has successfully solved the challenge. If the value is not correct, the student will be able to see the effects and have an opportunity to reassess her preconceptions about the underlying model.

To force the student to really think about it and do the actual operations involving the equations (and thus, learning the physic laws) and not only to try different random values until he finds out the right one, there's only one chance in every phase. When a phase ends, a new phase with random values for the parameters begins, or the mini-game ends.

B. *Stop that Box!*

The challenge in "Stop that box!" is to stop a sliding box in a given area of an inclined platform. The game is based on force

ISBN: 978-84-8158-561-2

and friction theories, and can be solved by the application of the equations of uniformly accelerated linear motion and other physical concepts. Again, result of the game is dependent on four parameters, the box mass (m), the box coefficient of friction (μ), the distance from the goal area to the box (x), the initial velocity for the box (v_i) and the angle formed by the platform with the horizontal (β) (Fig. 6). In this mini-game the concept of game phases also applies, meaning that even if the player is given multiple attempts the value will need to be recalculated for each one to prevent the student from doing random attempts.

In a typical game phase, the player would be given the values for some variables (m, x, and β, in the example) and would be asked to provide a value for the missing one (μ, in the example). If the player provides a value that is too small, the box will slide through the target area, while a value too big and the box will stop too soon. The player will actually see the consequence of this, thanks to the simulation. This same mini-game can be easily adapted and used to explain the friction concept because if there is no friction (that is reflected in μ=0) even if the initial velocity is 0 the block will move (the block can only stay in one place due to friction).

C. *Balance the Units*

The goal of this game is balancing a scale. To solve it the student should apply the knowledge about unit conversion (Fig. 7). At the beginning of the phase, a weight with a random value and a random unit appears on the left side of the scale. Over the right side, it appears some others weights with different values and units.

The purpose of the game is to choose the right weights from the right side to equal the weight in the left side. To do that, the player drags the chosen weights into the selected area. When the player thinks he has the right weights selected and clicks "GO!", the physic simulation starts and the left weight and the selected weights falls over the balance.

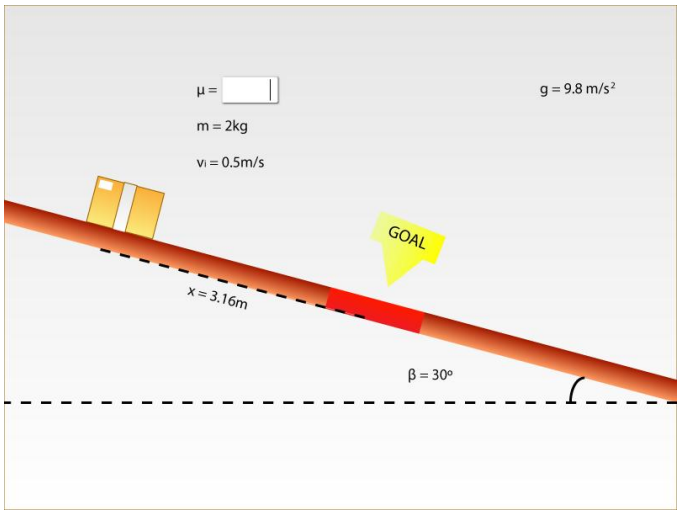

Fig. 6. In this phase of "Stop that box!" the unknown parameter is the coefficient of friction. The player must provide a value that forces the box to stop sliding at the goal area.

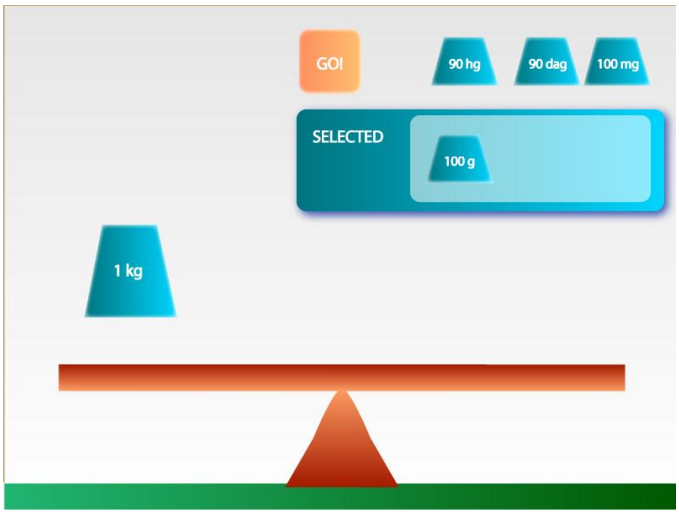

Fig. 7. In the "Balance the units" game, the player has to apply his knowledge in unit conversion to balance a scale.

If the scale stays in equilibrium for three seconds, the phase is correctly completed and the player has successfully solved the challenge.

V. Finishing Remarks

In this paper we describe our ongoing work on the combination and integration of a physics engine to extend the possibilities of the adventure games created with eAdventure. In this approach, we are only taking advantage of some of the features provided by the Box2D 2D physics engine to increment the range of mini-games that can be included into eAdventure games. The game examples presented are based on actual physics exercises for students between 14 and 16 years old. These examples and other potential physics games attempt to exploit both the beneficial effects of visualization and dynamic simulation in physic education with the engagement provided by the game environments.

Physics engines in general and Box2D in particular have many other features that can be exploited from an educational point of view (e.g. restitution or complex physical structures, like pulleys and other mechanical constructions). All this elements can be used to target other subjects in typical physics education using the games (or at least the game-like strategies) to promote the acquisition of the require concept or the practice of a skill (e.g. learning the formula, calculation of results). Thus the engine can be used to reinforce concepts as the examples presented can be developed using them.

A simplified edition, through the use of the described mini-games, provides support of the physics engine targeted at non-expert users. The full range of functions in the engine, however, is available for programmers and experts users to further increase the games that can possible be created using the eAdventure game platform.

Future work includes the creation of more of these potential mini-games, as well as the use of physics to increase the interactivity of environments in adventure games (e.g. make

Martín Llamas Nistal, Inmaculada Plaza García, Carlos Vaz de Carvalho, editores
ISBN: 978-84-8158-561-2

things fall, or the player slightly affect the environment, increasing its visual appeal). Moreover, the evaluation of physics mini-games with real students is important to establish the validity of the model and to find new mini-games that might reinforce other concepts.

ACKNOWLEDGMENT

The Ministry of Education (grant TIN2010-21735-C02-02) and the Ministry of Industry (grants TSI-020110-2009-170, TSI-020312-2009-27) have partially supported this work, as well as the Complutense University of Madrid and the Regional Government of Madrid (research group GR35/10-A and project e-Madrid S2009/TIC-1650), and the PROACTIVE EU project (505469-2009-LLP-ES-KA3-KA3MP) and the GALA EU Network of Excellence in serious games (FP7-ICT-2009-5- 258169).

REFERENCES

[1] L. A. Annetta, J. Minogue, S. Y. Holmes, and M.-T. Cheng, "Investigating the impact of video games on high school students' engagement and learning about genetics," *Computers & Education*, vol. 53, no. 1, pp. 74-85, Aug. 2009.

[2] R. Garris, R. Ahlers, and J. E. Driskell, "Games, Motivation and Learning: A Research and Practice Model," *Simulation & Gaming*, vol. 33, no. 4, pp. 441-467, 2002.

[3] Richard Blunt, "Does Game-Based Learning Work? Results from Three Recent Studies," *Training*, 2007. [Online]. Available: http://patrickdunn.squarespace.com/storage/blunt_game_studies.pdf. [Accessed: 27-Oct-2011].

[4] P. Felicia, "Digital games in schools: A handbook for teachers," in *European Schoolnet*, 2009.

[5] M. Pivec and P. Pivec, "Games in Schools," *Learning*, 2008. [Online]. Available: http://games.eun.org/. [Accessed: 27-Oct-2011].

[6] L. P. Rieber, "Seriously considering play: Designing Interactive Learning Environments based on the Blending of Microworlds, Simulations and Games," *Educational Technology Research and Development*, vol. 44, no. 2, pp. 43-58, 1996.

[7] A. McFarlane, A. Sparrowhawk, and Y. Heald, "Report on the educational use of games," in *TEEM: Teachers Evaluating Educational Multimedia*, 2002.

[8] S. Wexler, K. Corti, A. Derryberry, C. Quinn, and A. V. Barneveld, "The eLearning Guild : Immersive Learning Simulations 2008 : Research Library," 2008. [Online]. Available: http://www.elearningguild.com/research/archives/index.cfm?id=128&action=viewonly. [Accessed: 30-Oct-2011].

[9] S. B. Issenberg, W. C. McGaghie, E. R. Petrusa, D. L. Gordon, and R. J. Scalese, "Features and uses of high-fidelity medical simulations that lead to effective learning: a BEME systematic review," *Medical Teacher*, vol. 27, no. 1, pp. 10-28, 2005.

[10] A. J. Faria, "The Changing Nature of Business Simulation/ Gaming Research: A Brief History," *Simulation & Gaming*, vol. 32, no. 1, pp. 97-110, Mar. 2001.

[11] P. Moreno-Ger, J. Torrente, J. Bustamante, C. Fernández-Galaz, B. Fernández-Manjón, and M. D. Comas-Rengifo, "Application of a low-cost web-based simulation to improve students' practical skills in medical education," *International Journal of Medical Informatics*, vol. 79, no. 6, pp. 459-67, Jun. 2010.

[12] K. D. Forbus, "Using Qualitative Physics to Create Articulate Educational Software," *IEEE Expert Intelligent Systems And Their Applications*, vol. 12, no. 3, pp. 32-41, 1997.

[13] C. B. Price, "The usability of a commercial game physics engine to develop physics educational materials: An investigation," *Simulation & Gaming*, vol. 39, no. 3, pp. 319-337, Jul. 2007.

[14] J. Torrente, Á. Del Blanco, E. J. Marchiori, P. Moreno-Ger, and B. Fernández-Manjón, "<e-Adventure>: Introducing Educational Games in the Learning Process," in *IEEE Education Engineering (EDUCON) 2010 Conference*, 2010, pp. 1121-1126.

[15] E. J. Marchiori, Á. Serrano, J. Torrente, I. Martínez-Ortiz, and B. Fernández-Manjón, "Extensible multi-platform educational game framework," in *Proceedings of The 10th International Conference on Web-based Learning (ICWL 2011)*, 2011.

ISBN: 978-84-8158-561-2