CONSTRUCTION OF VIDEOGAMES BY ENGINEERING STUDENTS FOR UNDERSTANDING MODELLING PROCESSES

Angel Pretelín-Ricárdez[1,2], Ana Isabel Sacristán[2]

Abstract

We present some results of an ongoing research project where university engineering students are asked to construct videogames where they need to use models of physical systems; the purpose of this is to help them learn more about modelling processes and the elements and concepts involved. That is, a constructionist approach of game design is used to promote modelling learning. In the case presented in this paper, 12 students, in their last year of studies, were asked to build a videogame where water is used to solve different levels (puzzles) of the game –and thus requires a model of liquid water behaviour that also needs to meet the restrictions imposed by the game engine. Through this videogame-building task, we observed that students, in a fun and engaging way in which they were receptive and open to experimentation, as well as to making mistakes and learning from those and from other students, were able to deepen and refine how they conceive the process of mathematical modelling, as was evident in written work and group discussions at the end of the activities.

Keywords modelling, videogames, constructionism, model-eliciting activities (MEAs)

1. Introduction, background and theoretical framework

Learning modelling is an important activity in the life of an engineering student, because it is the first step to develop the design of any device, system, product or process. It requires students to apply mathematical skills in order to build appropriate representations and models.

In this paper we present some results from an ongoing project where university engineering students are asked to build videogames that include models of physical systems, in order to help them learn about modelling processes and understand the concepts and elements involved. In this case, we focus on a videogame that involved, in its creation, modelling liquid water behaviour.

The idea of asking students to build a videogame for learning about modelling is based on the constructionist paradigm: Constructionism [1] considers that learning is facilitated when the learner engages in the active construction and sharing of external objects; it emphasises the importance of providing learners with opportunities and a context to build hardware or software artifacts that engage them and are meaningful to them. Thus, in our quest to help students learn about modelling,

[1] Instituto Politécnico Nacional, UPIITA, Av. IPN 2580, 07340, Mexico City, DF, Mexico

[2] Centro de Investigación y de Estudios Avanzados (Cinvestav-IPN), Department of Mathematics Education, Av. IPN 2508, 07360, Mexico City, DF, Mexico.

we thought of engaging our engineering students in a construction activity –that of programming a videogame– that would require modelling.

In connection with the above, since the early days of computers in education, game design and programming have been proposed as a methodology for engaging learners, as in the case of the work of Harel [3]. In particular, the works of Kafai and colleagues, e.g. [4], [5], [6] and [7], present some of the foundations and structure of this methodology, and their relationship with the constructionist paradigm, even though those works are with younger learners than the ones in our study. One of the key ideas is that:

> Game design provided a situation that naturally combined issues of practice and theory, and reflection on those relationships and game design provided opportunities for discussion, reflection, and collaboration within a meaningful context. [6, p.180]

Following on the idea of providing students with, we also base our work on the concept of microworlds, which fits well with our proposal to engage students in the programming of a videogame.

In our work, the programming activities of the game design were conceived within a learning microworld, which we considered a suitable context for engaging in the activity of "building a meaningful product." Hoyles & Noss [2] define a microworld as composed of four components: the student component (concerned with the existing understandings and partial conceptions that the learner brings to a learning situation); the technical component (constituted by the software or programming language and a set of tools which provides the representational system for understanding a mathematical structure or a conceptual field); the pedagogical component (the didactical materials and interventions that take place during the computer-based activity); and the contextual component (the social setting of the activities).

Since we want students to go thru, explore and think about modelling processes, the programming activities of the videogame – which are part of the pedagogical component of the microworld – were conceived as a set of what Lesh and Doerr [8] have named *model-eliciting activities* (MEAs) which they define as problem–solving activities that are

> so called because the products that students produce go beyond short answers to narrowly specified questions – which involve sharable, manipulatable, modifiable, and reusable conceptual tools (e.g., models) for constructing, describing, explaining, manipulating, predicting, or controlling mathematically significant systems. [8], p. 3]

MEAs involve six principles of design [9], [10]: Reality, model construction, model documentation, self evaluation, model generalization, simple prototype. We took into consideration those six principles to integrate the modelling process, in this case of water behaviour, as part of a sequence of activities (see next section) that lead to the programming of the videogame.

2. Research aims and methodology

In our activities, we seek, through the videogame design and programming, for students to construct and reconstruct how they conceive and put into practice the modelling processes. The activities were carried out as part of an optional course that we created, for engineering students, at

the National Polytechnic Institute in Mexico City, Mexico.

We began with an the initial questionnaire, where students had to express and reflect on what it means, in general terms, to model and the general processes involved in it, simulation and videogame construction. For example, we asked students to define, in general terms, modelling, what a model was, what a simulation was among other questions. In particular, the main question was: *What steps are needed to build a model?*. Their written answers gave us, as researchers, and insight into their conceptions, with regard to the modelling process, before and after the activities and allowed us to observe if and how students transformed their conceptions through the activities.

During the activities, participants were organized into teams of two, within a learning microworld. We explain our methodology in terms of each of the components [2] of this learning microworld.

2.1. The technical component

Within the technical component, we have the game engine. In the examples given in this paper, we used GameMaker Studio [11] and its physics engine Box2D [12], through which students built a videogame that used the mathematical model of water to simulate this element in a virtual environment governed by the basic physical laws of the real world. These technological environments are meant to be rebuildable and modifiable products.

2.2. The student component

The students involved in the case presented in this paper, were 12 engineering university students of the National Polytechnic Institute, in their 8th semester of studies. They voluntarily signed up for this optional course, because they are interested in engaging in videogame development after college. It is important to mention that these students had previously taken at least one course in classical modelling and simulation.

2.3. The pedagogical and contextual components

The pedagogical component includes a sequence of three activities designed taking into account the six principles, mentioned above, of MEAs [9], [10] (see Table 1) to encourage students to construct and reconstruct concepts about mathematical modelling of physical systems.

After the reflection in the initial questionnaire, the first activity was work on to the actual idea and collaborative design of the videogame. Activity 2 required each student to individually propose a mathematical model And Activity 3 was again collaborative work to solve the actual modelling problem and incorporate a modified (adapted and simplified) version of the mathematical model into the videogame. Thus, in terms of the context, the activities (1 & 3) combined collaborative work in pairs, but also individual work (activity 2 –also the written questionnaires where answered individually). In this way students discussed and worked on their proposed solutions about the modelling of the water behaviour, developing a genuine engagement in the creation of the conceptual design of the videogame. We now describe each of these activities in more detail and in Table 1, we illustrate how the design principles of the MEA's apply to each of the activities.

Activity 1. In this activity, each team of students had to develop an idea for their videogame through a brief "story board" of up to 4 images, and adding a brief description of the gameplay. We required that the gameplay used water as the main element to solve puzzles and complete levels.

This is a task meant to genuinely engage students in creating a meaningful product.

Activity 2. This was an individual activity, where students had to build a model of water behaviour, that later could be used in the programming of the videogame designed in activity 1. The creation of this model requires students to have, and use, previous knowledge of algebra, vector calculus, differential calculus and also of fluid mechanics (statics and dynamics of fluids). The task is presented, not as a traditional problem statement, but as an open problem, as it happens in real life.

Activity 3. Here students had to collaboratively program the game using the models that they created (i.e. discussing the best way to apply one of them, or defining together a new one), into the game engine (in this case, Game Maker Studio [11]), thus rendering those models more concrete and meaningful. This requires going thru a stage of refining and simplifying the model, proposing appropriate restrictions and relationships (e.g. of scale, metric system, degree of freedom, etc.); as well as defining the characteristics of the computational objects that will simulate the particles of water, giving them attributes (shape, colour, and size) and some physical properties (gravity, density and potential and kinetic energy) represented by algebraic mathematical equations.

MEA design principle	Related activities
Model documentation	In the questionnaires, students already have to describe, for a general and abstract case, the steps of building a model. In Activity 1, the story board is a first documentation of the model. Then, in Activity 2, students have to describe the mathematisation of the water model. And a final documentation is found in the actual program, in the game engine, of the videogame.
Reality	The problem of designing (Activity 1) and programming (Activity 3) a model of water behaviour is realistic problem. Furthermore, constructing a real videogame is something that genuinely engages students in the task of mathematical modelling.
Model construction	Students build a first version of the mathematical model (Activity 2) and they make a refinement and simplification of this model when having to adapt it for, first, constructing a simulation in the game engine (Activity 3), and then implementing it in the actual videogame. Thus, students need to have clear understanding of the relationships necessary to program the simulation and, later, the videogame.
Simple prototype	When students set appropriate relationships and constraints for establishing a model with simplified equations (Activity 3), they can also provide simple and structured steps to generate a set of tests, such as the simulation of the model, in order to test its effectiveness.
Model generalization	As described in the previous point, in Activity 3, students need to establish appropriate restrictions and relationships that lead to a general model that can be used again in other situations and with other technological tools.
Self evaluation	During the stage of refining and simplifying the model (Activity 3), students need to evaluate their models and thru discussions take decisions for finding an implementable version of the model into the game engine. Also, thru the post questionnaire, a reflection is carried out about what was involved in the modelling process and how the entire process of the videogame creation, transformed and enriched their concepts from their initial conception (as given in the initial questionnaire).

Table 1: Relationship of the six principles of MEAs and the activities carried out by students.

3. Sample results

We now illustrate the unfoldment of each of the above activities, thru a case study of two students –

Felipe and Javier– who worked as a team.

3.1. The initial questionnaire

In the Table 2, we give the replies that these students gave to the question: *What steps are needed to build a model?*

Student	Question / Answer
	What steps are needed to build a model?
Felipe	6.- Movimientos, variables y (grados de libertad) Movements, variables, and (degrees of freedom)
Javier	Analizar el fenomeno, representarlo matematicamente, analizar su comportamiento, implementar una accion en base a lo anterior. Analyse the phenomenon, represent it mathematically, analyse its behaviour, implement an action based on the above.

Table 2: Question / answers of the Initial questionnaire.

We observe that Felipe only mentions some aspects to take into account and doesn't give actual steps needed to build a model; Javier has a more clear and structured (although perhaps incomplete) idea. Previously, as mentioned above, the students had studied courses on modelling, so they know how to build models in practice; but, as seen here, they have difficulties in expressing the steps needed to build a model.

3.2. Activity 1

In this activity, students constructed a brief "story board" (see Figure 1) to describe the gameplay. The main idea of Felipe and Javier's game is to present the player and two containers (one empty and one filled with water). These containers can appear anywhere in the playing area. The player must take the water of the filled container, and fill the empty one, aided by gravity or other resources that may appear in the play area. Felipe and Javier, as did all the other students, showed great interest in this activity, because they want to engage in videogame design when they graduate.

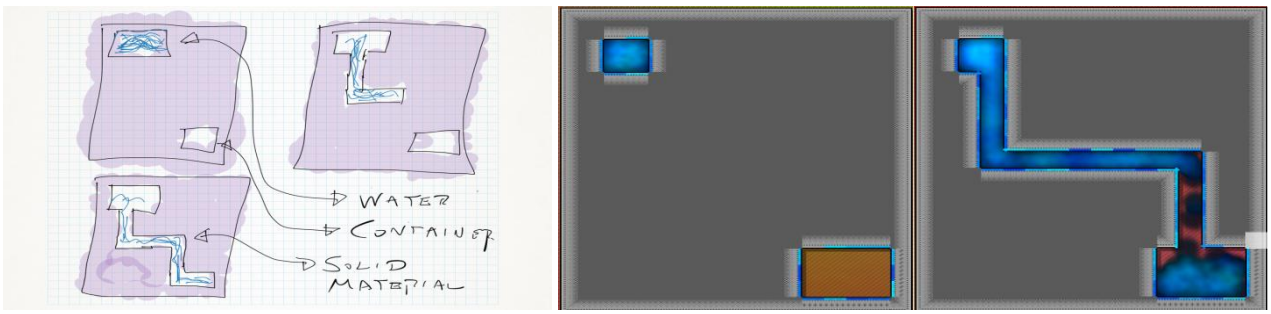


Figure 1: Videogame's storyboard (left) with screenshot of the resulting videogame (right).

3.3. Activity 2

As explained above, in this activity students had to construct a mathematical model of the water behaviour, that included realistic physics, for implementing it in the previously designed game.

Each student built a mathematical model of fluid mechanics, addressing the water model either as a molecular model (Felipe's case) or as a continuous model (Javier's case) –see Table 3.

Felipe's model (molecular model)	Javier's model (continuous model):
Felipe built a, mathematical model, based on, intermolecular interaction force. $\overrightarrow{F_{12}} = -\frac{\partial e_{12}(d)}{\partial \overrightarrow{r_1}} = 6\frac{e_0}{d_0}\left[2\left(\frac{d_0}{d}\right)^{13} - \left(\frac{d_0}{d}\right)^{7}\right]\frac{\overrightarrow{r_1} - \overrightarrow{r_2}}{d}$ Where: e_0 **,Finite distance at which the inter-particle potential is zero (potential well)** $d_0 \approx 3 \times 10^{-10} m$**, Critical distance** $e_{12} = e_0\left[\left(\frac{d_0}{d}\right)^{12} - \left(\frac{d_0}{d}\right)^{6}\right]$**, Potential energy of interaction between molecules (Lennard-Jones Potential)** $d = \lvert\overrightarrow{r_1} - \overrightarrow{r_2}\rvert$**, Intermolecular distance between two molecules**	Javier built a, mathematical model, based on, water macroscopic behavior. $\rho(\vec{r},t) = \lim_{\Delta V \to 0} \frac{\sum_{i=1}^{N(\Delta V)} m_i}{\Delta V}$**, Local density** $\vec{v}(\vec{r},t) = \lim_{\Delta V \to 0} \frac{\sum_{i=1}^{N(\Delta V)} m_i \overrightarrow{v_i}}{\sum_{i=1}^{N(\Delta V)} m_i}$**, Local speed** $E(\vec{r},t) = \lim_{\Delta V \to 0} \frac{\sum_{i=1}^{N(\Delta V)} \frac{1}{2} m_i \lvert\overrightarrow{v_i}\rvert^2 + \frac{1}{2}\sum_{i=1}^{N(\Delta V)} \sum_{j=1(j\neq i)}^{N\Delta V} e_{ij}}{\sum_{i=1}^{N(\Delta V)} m_i}$**, Local energy** $\frac{E(\vec{r},t)}{\sum_{i=1}^{N(\Delta V)} m_i} = \frac{1}{2}\lvert\vec{v}(\vec{r},t)\rvert^2 + u(\vec{r},t)$ **, Specific energy** Where: $N(\Delta V)$**, The number of molecules contained in the volume** ΔV **at time "t"** $\vec{r}$**, It is the position vector of the point where you are considered local properties** m_i**, It is the mass of a given molecule** $\overrightarrow{v_i}$**, It is his molecular speed** e_{ij}**, It is his potential energy** u**, The internal energy contained in the volume element**

Table 3: Felipe's and Javier's mathematical models.

3.4 Activity 3

Here the two students shared their mathematical models and, in discussing them, noted that it would not be possible to simulate them for validation in the game engine (Game Maker Studio), concluding that they must simplify them.

3.4.1 Simplifying the model

For constructing an appropriately simple model, students had to take into account certain constraints and relationships of the physical engine of GameMaker Studio, such as the metric system, scales, and that instead of the model being applied to a real 3D environment, it had to work in a virtual 2D world. This implied adapting the formulas. More specifically:

Because the game took place in a 2D environment, students had to be aware that the bodies were constrained in their movement to three degrees of freedom: i.e. two translational coordinates and one rotational coordinate.

The students observed that the physics engine (inside the game engine) uses as measuring system the metric one, but limits objects to have a size equivalent to between 0.1 m and 50 m; however it also sets that pixels should not be in a one-to-one relationship with meters (i.e. one must not set the equivalence of 1:1 meter to pixel), and radians are used as angles. Thus, the scaling ratio used by Felipe and Javier, inside the virtual physical world, was 10:1 (i.e. 10 pixels equal 1 m). This relationship was expressed through the code: *physics_world_create(1/10)*. Using this relationship, the students created virtual objects inside the physical world: e.g., water was represented as a set of circular-shaped particles with a radius equal to 4 pixels (i.e. equivalent to 0.4 m).

Another important relationship in the simulation was to establish the value of the constant of gravity acceleration (g). For this, the students decided that the constant, which is equal to 9.8 m/s^2 ,

should be rounded to 10 m/s^2, due to the scaling factor that they needed to use; this meant that the acceleration would be of 100 pixels per second, every 60 steps, in the direction 270°. This was expressed through the following segment of code: *physics_world_gravity(0, 10);* and with a *room_speed* of 60. The value of g affects every object in the virtual world differently, depending on the properties of that object: friction coefficient, linear damping, angular damping, and density.

To set the value of density for each water particle simulated in the virtual world, students had to take into account that the density used was not a volumetric density one (3D), but a surface density (2D): that is, ρ=1000 Kg/m^2 (with square meters, instead of cubed as in ρ=1000 Kg/m^3). This is expressed in code as: *physics_fixture_set_density(Water_particle, 1000)* where *Water_particle* is the name of the object representing a particle of water and 1000 is the value of water density.

Having established these relationships to the density, students established other relationships, such as the friction coefficient between the water particles. In GameMaker Studio, friction is defined as the force that resists movement or slippage of a material over another, as can happen when two materials collide in the virtual physical world. The friction value must be between zero and one. In this case, because they were modelling water which is a fluid, the students tried to define this coefficient as a coefficient of viscosity expressed in Kg/m*s, giving it a value of 0.01, considering that the viscosity value between water particles is very close to 0. This is expressed as:

physics_fixture_set_friction(Water_particle, 0.01);.

Another property that students had to define, is the restitution coefficient that indicates the amount of kinetic energy that remains after a collision between particles or objects; that is, this coeffcient refers to the level of "bounce" between particles. The restitution value is between zero and one. The water in the real world has a coefficient of restitution of 0.98 and in the virtual world, Felipe and Javier assigned it the value of 0.9, which is expressed as:

physics_fixture_set_restitution(Water_particle, 0.9);.

Finally, the students had to establish relationships for the linear and angular damping, which refers to the forces that oppose the linear and rotational (angular) motion of a particle (like air friction). Air friction dampens water motion. A simplified representation of air friction is the formula Af = -K*v, where K is the damping coefficient dependant on the shape of the body, and v is the velocity of each particle, dependent on its interactions in the virtual world. Damping forces are normally expressed with values between zero and infinity, but the game engine uses a simpler method, giving shapes that have less air resistance (i.e. round ones) values close to zero and rectangular shapes that have more resistance, values near 1. Thus, in this case, since water particles are round, students assigned the value of 0.1 for the damping coefficient. This is expressed in code as:

physics_fixture_set_linear_damping(Water_particle, 0.1);
physics_fixture_set_angular_damping(Water_particle, 0.1);

Having defined the relationships, constraints and properties, as shown above, students were able to develop a water model simulation.

3.4.2 The simulation

Felipe and Javier concluded that it was more feasible to build a model of the water behaviour, based on a simplified molecular model –where each particle of water is affected by the following factors: specific gravity ($\gamma=\rho \bullet g$), density ($\rho=m/V$), kinetic energy ($Ec=1/2 \bullet m \bullet v2$) and potential energy ($Ep=m \bullet g \bullet h$)– instead of building a continuous model (as in the macroscopic point of view), because

of the restrictions of the game engine.

They then conducted tests to validate the model, including simulations to show the Archimedes' principle which relates to the upward buoyant force that is exerted on a body immersed in a fluid (Figure 2) as well as other tests (e.g. for Pascal's principle, which says that any change in pressure in any point of a resting enclosed fluid, is transmitted undiminished throughout the fluid). With the validated model, it was relatively easy for the students to then program the game they had designed. A screenshot of the videogame can be seen in Figure 1.

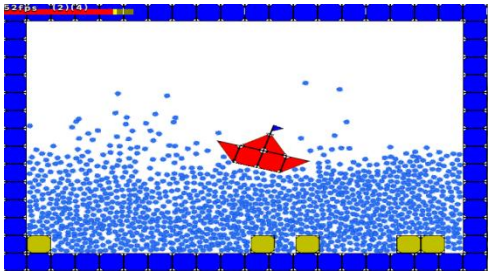


Figure 2: Simulation of Archimedes' principle.

3.5 The post-questionnaire

The post-questionnaire focused on the question: *What steps are needed to build a model*? requiring to draw a block diagram for the steps given. Table 4 shows Felipe's and Javier's answers. We can observe the evolution of the conception of the modelling process as compared to the answers in the initial questionnaire. However, in Felipe's case, from his diagram, we see that he doesn't conceive the modelling process as an iterative activity; in contrast, Javier's block diagram (without taking into account if his representation is correct or not) includes feedback loops in the internal processes.

Student	Answer / question
	1.- *What steps are needed to build a model?*
Felipe	First I analyze the problem. Then I see what I have. I investigate the characteristics and properties of that which will be modelled. I see if the technological tools that I have, allow me to make the model. Primero analizo el problema Despues veo que es con lo que cuento investigo las caracteristicas y propiedades de lo que valla a modelar Ver si las herramientas que tengo me permiten realizar el modelo
Javier	1.- Conceptualize the system to be modelled. 2.- Identify the variables that affect the system. 3.- Mathematise the system with all the variables. 4.- Test the model without disturbances. 5.- Test the model taking into account the disturbances. 6.- Improve the model if necessary. 1=Conceptualizar el sistema a modelar 2= Identificar las variables que afectan al sistema 3= Matematizar el sistema con todas las variables 4= Probar el modelo sin perturbacion alguna 5= Probar el modelo contemplando las perturbaciones 6= Mejorar el modelo en caso necesario.

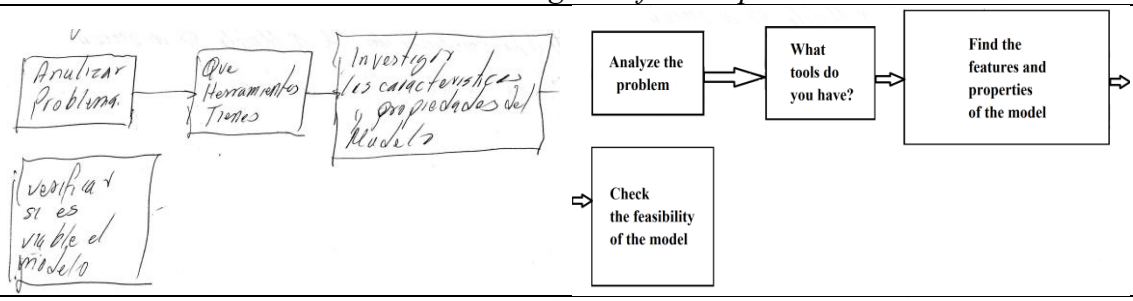
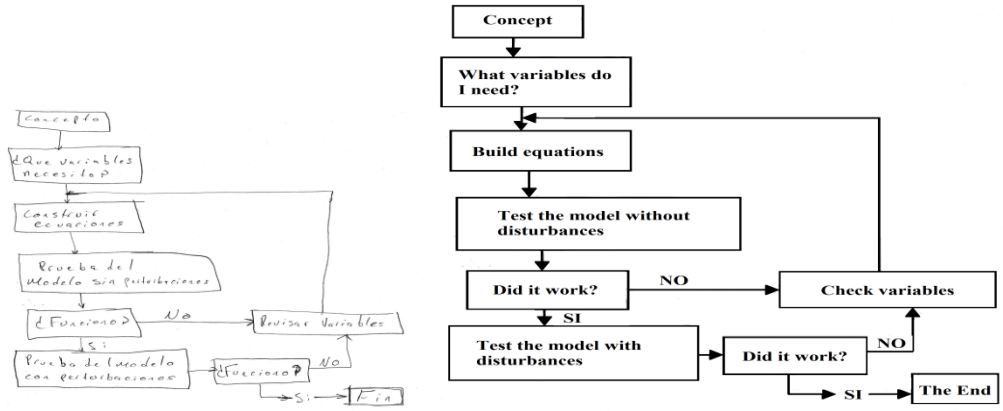
	2.- Draw a block diagram of the steps above
Felipe	
Javier	

Table 4: Questions / answers of the post-questionnaire.

3.6 Reflections and discussions on the process

After the activities, students engaged in a group discussion and deep reflection about the process of modelling in engineering work. They commented on how their previous teachings had given them an incorrect conception of what is involved in a modelling process, but now understood a modelling process, as an iterative and perfectible activity, replacing the traditional concept of "problem statement" requiring a fixed, final answer. Thus, thru the MEAs and the experience of building the videogame, students transformed and enriched the way they conceive a modelling process.

4. Final remarks

The activities faced students with a real problem (the design and programming of a videogame) that is significant in its sociocultural context. Thru the programming of the videogame, students engaged in producing a working model that was meaningful to them and gained a deeper understanding of all the elements involved in the modelling process. The mathematical model of water behaviour requires students to apply differential calculus (partial derivatives) and vector calculus. But, what happens when the equations cannot be simulated with the technological tool being used (in this case, the game engine)? This is the question that gives meaning to the proposed activities: thru the restrictions imposed by the game engine, we lead the students to adapt the model, as well as rethink their ideas about the modelling process. Students not only had to produce mathematical representations of the physical phenomena (water behaviour), but also understand these in the context and dynamics of the game they were building, establishing relationships between appropriate variables and restrictions (imposed by the technological restrictions of the game engine), and physical formulas in order to adapt their mathematical model. Specifically, they had to establish relationships between the computational objects (in this case water particles) – which also had to be attributed values such as form, colour and size–, and their physic-

mathematical properties (gravity, density and potential and kinetic energies). Students were surprised by the great challenge that this was (they had assumed it would be much easier), especially in terms of working with scales and finding the relationships, but learned from their mistakes and thoroughly enjoyed it as well as working and learning from their peers. This constructivist process also helped them appreciate how –unlike traditional problem statements in class that look for fixed answers– in the real world, modelling processes are cyclic (iterative) and perfectible, and often collaborative. We now continue work with a new group of students (but including two students from the above experiences) to simulate robotic manipulators using another 3D game engine (Unity) that is more robust and complete.

References

[1] S. Papert and I. Harel, "Situating constructionisn," in *Constructionism*, S. Papert and I. Harel, Eds. New Jersey: Ablex Publishing Corporation, 1991.

[2] C. Hoyles and R. Noss, "Synthesizing mathematical conceptions and their formalization through the construction of a Logo-based school mathematics curriculum," *Int. J. of Math. Educ. In Sci. and technology*, vol. 18, no. 4, pp. 581-595, 1987.

[3] I. Harel, I, "Children as Software Designers: A Constructionist Approach for Learning Mathematics," *J. of Math. Behavior*, vol. 9, no. 1, pp. 3-93, 1990.

[4] Y. B. Kafai, *Minds in play. Computer game design as a context for children's learning*. Hilsdale, NJ: Lawrence Erlbaum Associates, 1995.

[5] Y. B. Kafai and M. Resnick, *Constructionism in practice: Designing, thinking, and learning in a digital world*. Mahwah, NJ: Lawrence Erlbaum Associates, 1996.

[6] Y. B. Kafai *et al*., "Game design as an interactive learning environment fostering students' and teachers' mathematical inquiry," *Int, J. of Comput. for Math. Learning*, vol. 3, no. 2, pp. 149-189, 1998.

[7] Y. B. Kafai, "Playing and making games for learning: Instructionist and constructionist perspectives for game studies," *Games and Culture*, vol. 1, no. 1, pp. 36-40, 2006.

[8] R. Lesh and H. M. Doerr, "Foundations of a models and modelling perspective on mathematics teaching, learning, and problem solving," in *Beyond constructivism. Models and modelling perspectives on math. problem solving, learning and teaching*, R. Lesh and H. M. Doerr, Eds. Mahwah, NJ: Lawrence Erlbaum Associates, 2003, pp. 3-33.

[9] R. Lesh *et al*., "Principles for developing thought-revealing activities for students and teachers," in *Research design in math. and Sci. Educ.*, A. Kelly and R. Lesh, Eds. Mahwah, NJ: Lawrence Erlbaum Associates, 2000, pp. 591-646.

[10] E. Hamilton *et al*., "Model-Eliciting Activities (MEAs) as a Bridge Between Engineering Education Research and Mathematics Education Research," *Advances in Eng. Educ.*, vol. 1, no. 2, pp. 1-25, 2008.

[11] Yoyo Games. (2014, March 07). *Game Maker Studio* [Videogame engine]. Available: http://www.yoyogames.com/gamemaker/download

[12] E. Catto (2014, March 07). *Box2D* [Physics Engine for Games]. Available: http://box2d.org/about/