

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Designing a programming-based approach for modelling scientific phenomena

Gordon Simpson, Celia Hoyles & Richard Noss

London Knowledge Lab, Institute of Education, University of London, UK

Abstract

We describe an iteratively designed sequence of activities involving the modelling of 1-dimensional collisions between moving objects based on programming in *ToonTalk*. Students aged 13-14 in two settings (London and Cyprus) investigated a number of collision situations, classified into six classes based on the relative velocities and masses of the colliding objects. We describe iterations of the system in which students engaged in a repeating cycle of activity for each collision class: prediction of object behaviour from given collision conditions, observation of a relevant video clip, building a model to represent the phenomena, testing, validating and refining their model, and publishing it – together with comments – on our web-based collaboration system, *WebReports*. Students were encouraged to consider the limitations of their current model, with the aim that they would eventually appreciate the benefit of constructing a general model that would work for all collision classes, rather than a different model for each class. We describe how our intention to engage students with the underlying concepts of conservation, closed systems and system states was instantiated in the activity design, and how the modelling activities afforded an alternative representational framework to traditional algebraic description.

Keywords

Nature of technology (hardware/software): Communication; Computer; Knowledge-Based; Modelling;

Learning process: Collaboration; Group; Individual

Level and place of learning: Secondary

Research/evaluation paradigm: Illuminative; Qualitative

Content: 1-Dimensional Collisions; Design; Conservation Laws; Mathematics; Programming

Introduction

This article outlines the iterative design, prototyping and testing of activities and open-ended software tools that allow 13-14 year-old students to investigate and model collision phenomena. The key idea of this work was for students to make predictions about the behaviour of colliding objects in specific circumstances, study corresponding video clips of real-world collisions, program models to instantiate the observed behaviour, test and validate their models by running them, and finally publish their models on the web for comment and critique. The development and piloting of the activities in this domain have been collaboratively conducted by teams at the Institute of Education in London, and at the University of Cyprus in Nicosia, as part of the *WebLabs*¹ project

¹ We acknowledge the support of the European Union, Grant # IST-2001-32200. WebLabs is a three-year European research project on the use of programming and web-based collaboration in mathematics and science education. Our

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

(www.weblabs.eu.com). We focus on the design approach we have adopted and illustrate this with some episodes of the system in use.

Our approach is based on inviting students to use *ToonTalk* (www.toontalk.com) in order to create models that represent the state of their thinking about the phenomena under consideration, and to share their models with each other using a specially-designed web-based system, *WebReports* (www.weblabs.org.uk/wlplone), which we describe below.

ToonTalk is a fully functional, concurrent programming language that has an interface modelled in the style of a video game (Kahn, 1996; 1999). We have used ToonTalk as a programming system for quite young students to construct their own games (Noss, Hoyles, Gurtner, Adamson and Lowe, 2002) and we have argued that such work forms part of a more general challenge to design and build systems that encourage important facets of mathematical thinking (Noss, 2001; Hoyles & Noss, in press).

In ToonTalk, every programming structure is concretized as an animated cartoon object: robots stand for programs, birds for message sending, nests for message receiving, scales for comparisons, trucks for process spawning, and bombs for process termination. The programmer directly manipulates these objects using a virtual 'hand', or with tools such as the bicycle pump (for changing object size), magic wand (for copying) or vacuum

focus is on communities of young learners (10-14 years), engaged in collaborative modelling of mathematical and

scientific phenomena, across six EU countries. We also acknowledge the assistance of our colleagues at the University

of Cyprus during work on this topic.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

cleaner (for cutting, pasting and erasing), as shown in Figure 1. Programs are created by training a robot – directly leading it through the steps of a task it is required to perform. The robot remembers what it was trained to do, and can be generalised from the specific example on which it was trained to handle more general cases, without explicit need for variables (by vacuuming out the specific example). Needless to say, this style of programming is very different from that used in traditional text-based languages.

[INSERT FIGURE 1 HERE]

Alongside working on the development and extension of the ToonTalk programming language, we have developed a web-based collaboration system we call WebReports, which allows students to share and discuss not only their current thoughts, difficulties and conjectures, but working models that instantiate their ideas. Students compose online reports using a visual editor, which includes the facility to embed files such as pictures and java applets, and most importantly ToonTalk objects, as shown in Figure 2. The two systems have been tightly integrated; with a few mouse clicks a student can upload and include a ToonTalk model in their online report. The embedded object is shown as a picture, which is also a hyperlink to the actual ToonTalk code. Another student (perhaps in another country) who views the report can simply click the ToonTalk picture and a version of the model will be automatically downloaded and opened in the ToonTalk environment on their local computer. Students can also discuss one another's reports using the commenting functionality. A comment can be posted at the bottom of any webreport page, and comment authoring has the same full 'wysiwyg' style of editing as report writing. Comments can be posted as replies to other comments so that threads of

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

discussion can be created (in much the same way as in internet newsgroups), along with new ToonTalk models. This collaborative workspace, although an important facet of our work in the WebLabs project, is given only cursory treatment in this paper and is described in more detail elsewhere (e.g. Mor, Tholander & Holmberg, in press).

[INSERT FIGURE 2 HERE]

We start by framing our approach to modelling and 1D collisions within previous research in this area. We then consider the underlying physics that governs the behaviour of objects colliding in 1-dimension and break the collisions down into six classes, based on the relative velocities and masses of two colliding objects. We then describe iterative design experiments through which we developed a sequence of activities based on these classifications (see Cobb, Confrey, diSessa, Lehrer, & Schauble, 2003). We present some illustrative findings from student activities during the second experiment, and finish with a discussion of successes, limitations and future plans for a further iteration.

Collisions as a knowledge domain

Collisions are ubiquitous. Although we tend to think of unintentional collisions such as those involved in motor vehicle accidents, collisions are a natural event in everyday life. From the point of view of physics, a car crash, a child kicking a ball, and a finger striking a keyboard are, of course, all considered types of collision and as mathematically-sophisticated adults we would expect (where children may not) that there are invariants that describe these situations irrespective of their specificities. Collisions are, in fact, one

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

of the most basic phenomena in which a quantity is conserved, and conservation theorems play a fundamental role in physics (de Jong, Martin, Zamarro, Esquembre, Swaak, & van Joolingen, 1999).

In school, collisions are often used as a context to introduce conservation of energy and conservation of momentum laws. However, the key conceptual insights – whether mathematical or scientific – are often obscured by the necessity of algebraic manipulation as a means to solve the relevant equations. Thus, for many students algebra often becomes an end in itself, with the result that the regularities and invariants of the collision situation are lost. Studies of undergraduate students have highlighted this problem: while introductory physics students might be able to ‘solve for x ’, they have difficulty understanding why, when and how to use conservation principles, such as those of momentum and energy conservation, to study phenomena (Grimellini-Tomasini, Pecori-Balandi, Pacca, & Villani, 1993; Lawson & McDermott, 1987; George, Broadstock, Vazquez, & Abad 2000). One finding from these studies indicates that students often fail to appreciate why they do not need to analyse the interactions occurring during collisions, but rather have simply to apply conservation laws to the initial and final states of the interacting system (Grimellini-Tomasini et al., 1993).

Our position on students’ learning owes much to the work of diSessa (1988; Smith, diSessa, & Rochelle, 1994) who suggests that rather than identifying ‘misconceptions’ or ‘alternative’ ways of conceptualising laws at a macro level, it is more productive from a pedagogical perspective to focus attention on knowledge that is contiguous with students’ prior understandings. We believe that there are three important and related

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

mathematical/scientific ideas that underpin the domain of collisions. These are the ideas of a *closed system*, of *conservation* and of *system states*.

The laws of conservation can only be applied to a closed system: a closed system is one in which there are no external forces acting on the objects in the system. In the case 1-dimensional collisions, the two colliding objects form the system and it is implicit that there are no external forces acting to alter their states (e.g. velocities). The question of what constitutes the closed system requires an appreciation that there are laws of, for example, conservation that work within a closed system, and in fact a closed system is defined as a system precisely because there are such laws that work in it. These kinds of fundamental ideas are often left implicit in standard school approaches.

The idea of conservation means that *something* is invariant over time or throughout an interaction. Identifying invariants and variants and how they interact, is crucial to the modelling process in both mathematics and physics (Hoyles, Morgan, & Woodhouse, 1999). In the present case, these quantities are energy and momentum, derived quantities that are nowhere directly observable in the system. We are not so much interested in students learning what these specific quantities are or how they are conserved, but rather if (and how) they approach the *idea* of conservation as a part of their modelling activities.

The importance of the concept of *system states* is that within a closed system, the value of a conserved quantity is invariant over time, and as a consequence – most crucially (but often left implicit) – that intermediate states of the system can therefore be ignored. In the context of collisions, we arranged matters so that students would focus their attention on

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

the pre-collision and post-collision states of the system, and the details of the collision process itself could be treated as an instantaneous interaction that *only* changed the state of the two colliding objects. We did not seek to teach this idea explicitly: rather it formed the core of the implicit substrate that underlay our modelling approach.

We return to this topic in the discussion to evaluate whether we can make sense of students' work when exploring collision phenomena in terms of their engagement with these three fundamental concepts. For the moment, we note that traditional algebraic representations and methods associated with the teaching of conservation laws in general, and collision phenomena in particular, tend to be problematic for students, most notably as they tend to divert attention from these fundamental issues. In WebLabs, we devised alternative ways to represent mathematical and scientific knowledge, in order to avoid a premature focus on algebra. In this, we followed Sherin's (2001) conjecture that not only may program-based representations be easier to understand than algebraic representations, but that they also may afford different understandings of the physics content.

In order to simplify the knowledge domain of collisions, we decided to restrict our focus to 1-dimensional (1D) collisions. The goal was for students to build models of a range of 1D collision phenomena in ToonTalk, in some (but not all) cases making use of tools we had constructed in advance. By building, testing and evaluating the models themselves, our aims were that students would gain some understandings of what it means to model, to follow up conjectures, to generalise and specialise; and simultaneously to develop understandings of the behaviour of colliding objects. The aims around modelling were

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

predicated on our belief that the process of constructing models is an evolving activity that is central to scientific inquiry (Sherin, diSessa, & Hammer, 1993).

The modelling approach is to some extent characterised as an iterative and cyclical procedure in which the model is refined to more closely approach some aspect of a physical phenomenon (Constantinou, 1999; Louca, & Constantinou, in press). ToonTalk is particularly well suited to model collision events, as it has built-in *sensors/remote controls* both for detecting collisions between objects, and for controlling object speeds, and it affords a dynamic animated metaphor that allows students to watch collisions and monitor and manipulate the values of relevant variables. We should however stress that we hardly expected that participating in the modelling cycle in the context of collisions would lead students spontaneously to discover the conservation laws themselves: rather, we expected they might learn about some of the ideas that underlie the principles of conservation and system states. In fact, an important scientific insight that we expected students to appreciate was that conservation laws exist, and that the search for quantities that are conserved forms a major part of what distinguishes science from mere observation.

Methodology

We worked with a group of six students aged between 13 and 14 years attending a North London secondary school. Activities took place outside the standard school curriculum: students were selected by their teacher from an ICT class on the basis of interest they had shown during two introductory ToonTalk sessions. The modelling activities were spread over eight 50-minute weekly sessions, followed by a full-day workshop for group

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

reflection and discussion. Sessions were conducted by two researchers, group discussions were audio-recorded, and video recording was used during the full-day workshop.

Students could work in pairs or individually at the computer.

Around the same time, a group of twenty 13-14 year-old Cypriot students were also working on 1D collisions, in two 1.5 hour sessions per week with two researchers present. These students worked over approximately the same period as the London students, and some asynchronous interactions were possible (though, due to linguistic and technical problems, difficult).

We reiterate our focus on design: our student numbers are relatively small, and we are modest in our claims to have observed significant learning over such a short period of time. Nevertheless, we have transcribed video and audio data of each session where possible (a challenge in the reality of a classroom) together with written (web)reports posted by the students: these have been used to delineate episodes that illustrate the process of model construction, testing, discussion and reflection that took place during the activities. We based this process of delineation closely on our epistemological objectives for the activities, rather than on seeking compelling evidence of individual or group learning outcomes, although we will provide some pointers, in what follows, in the latter direction.

The physics of 1D collisions

We begin by considering the physics that governs the behaviour of two colliding objects.

Figure 3 shows the *before* and *after* collision states of a two-cart system with the mass

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

and velocity variables labelled. Note that velocity is a vector quantity that has been defined as positive if movement is towards the right; a negative velocity signifies movement towards the left. To simplify the situation still further, the collisions are assumed to be perfectly elastic (there is no loss of energy), and friction is ignored (naturally, it was expected that discussion of these assumptions would become an important part of the interactions between students, and between students and the teacher).

[INSERT FIGURE 3 HERE]

This situation is governed by two laws: conservation of momentum and conservation of energy. The law of conservation of momentum states that the total momentum of a system does not change if there are no external forces. Momentum is defined as mass times velocity, so in our case:

$$m_1.v_1 + m_2.v_2 = m_1.v_1' + m_2.v_2' \quad (1)$$

Conservation of energy states that in a closed system total energy does not change. In our case we are dealing objects in motion, which is kinetic energy:

$$\frac{1}{2}.m_1.v_1^2 + \frac{1}{2}.m_2.v_2^2 = \frac{1}{2}.m_1.v_1'^2 + \frac{1}{2}.m_2.v_2'^2 \quad (2)$$

The conservation of energy and conservation of momentum equations can be combined and rearranged to give formulae for the post-collision velocities of the carts:

$$v1' = ((m1 - m2)/(m1 + m2)).v1 + (2m2/(m1 + m2)).v2 \quad (3)$$

$$v2' = ((m2 - m1)/(m1 + m2)).v2 + (2m1/(m1 + m2)).v1 \quad (4)$$

We now consider two special cases where the algebra can be simplified. First, if we consider the case where carts are of equal mass ($m1 = m2$), it is easy to see that the equations simplify to:

$$v1' = v2 \quad (5)$$

$$v2' = v1 \quad (6)$$

This means that the post-collision velocity of cart1 is equal to the pre-collision velocity of cart2 and vice versa – or put another way, the velocities swap. Thus for collisions between *same-mass* objects in 1-dimension, a *velocity swapping* model works for all cases.

Second, if we consider the case where one object is stationary and its mass tends to infinity ($v2 = 0$, $m2 \rightarrow \infty$), simple algebra shows that the equations simplify to:

$$v1' = -v1 \quad (7)$$

$$v2' = 0 \quad (8)$$

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

So in this case cart1 simply reverses direction after collision, while cart2 remains stationary, which models the situation when a moving object hits an immovable object, such as a wall.

We should emphasise that while this elementary algebra might clarify the situation for the reader, our objective was for students to begin to explore the situation without need of any algebraic facility and therefore to be able to assess the possibilities and potentialities of alternative ways to represent these relationships.

Classes of 1D collision

Consider the system shown in Figure 3 in which cart1 is to the left of cart2: the carts are on a collision path due to their horizontal velocities. We have defined six different classes of perfectly elastic collision that can occur between the carts², as shown in Table 1. The collision events in the different classes appear visually different, although they are all governed by the same laws of conservation of momentum and energy³. The first four classes are similar in that they deal with carts of the same mass. The fifth class represents

² A mirror image of the classes can also be defined by swapping and negating the velocities of cart1 and cart2. The axis of positioning and movement is also arbitrary, e.g. it could be in the vertical dimension.

³ One of the interesting things about the collision classification is that it relies on a stationary frame of reference. If the frame of reference is defined so that it is moving at the same speed as one of the carts, then the apparent distinction between the same-mass collisions (A-D) disappears. For example, if we specify the frame of reference as moving at the same speed as cart1, then all collisions appear as if a stationary cart1 is hit by an approaching cart2. It is possible to model this in ToonTalk in various ways. However, it was not clear how we could leverage this to aid learning, especially given studies that show frames of reference to be difficult for university level students to understand (Sherr, Shaffer, & Vokos, 2001).

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

collisions between carts of differing mass that can potentially be broken down into further subclasses depending on the relative mass and velocity of each cart. The sixth class is a particular case of the different mass class E, in which one cart has infinite mass (i.e. is immovable). Our goal was for students to engage with and explore all six classes of collision. Our first design decision was to determine how students should approach the modelling and in what order they should experiment with the different classes of collision.

[INSERT TABLE 1 HERE]

Our approach to answering these design questions was iterative, and is detailed in the following section. Specifically, we describe students' work in modelling the same-mass collision classes A-D, the special case class F, and present our rationale and future plans for modelling the different mass class E.

Development of an activity sequence for 1D collisions

When we first started planning activities and building tools in ToonTalk for students to investigate collisions, we tried to develop a general-purpose collision model, that is a model that would work for 2-dimensional collisions between any number of objects of varying masses, which could then be made specific for 1-D case. After prototyping, testing and discussion, we realised we were running the danger of obscuring the important parts of the model with the extra details required to make it general-purpose and flexible. We found, like other designers before us, that flexibility and generality were gained at the expense of transparency of the tools, and that students were liable to

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

become overly focused on the tools themselves, rather than the knowledge we intended they would investigate⁴. Our solution was therefore to restrict ourselves to modelling collisions in 1-dimension. We then derived the classification of the classes of collision as shown in Table 1 in order to help us understand the collision phenomena. Later, we revisited this classification in order to use it as the basis for the design of activity sequences for students.

Starting the sequence with one cart of infinite mass

In the first iteration of the design cycle, we started by considering the situation of a ball bouncing off a wall or the ground (class F in Table 1, and described algebraically in equations 7 and 8). In this scenario, the ball is a ‘movable’ object and the wall it collides with is immovable (or at least, can be thought of as not moving in terms of state change). We started with this collision situation because it can be easily demonstrated in the real world, seemed simple to understand and to approximate as a model, and was also straightforward to implement in ToonTalk. Because the modelling is simple, students could build models themselves without needing pre-built tools and hence class F promised to serve as a good introduction to the relevant ToonTalk programming devices. Our plan was thus for students to start with a system of one movable and one fixed object (class F), and then move on to consider a system with two movable objects (classes A-E).

⁴ For example, we built a model in which any number of objects could collide with one another. This required each of the objects to send information about their speed at collision to a processing module, that would then sort the information and apply an algorithm to it. Although this worked rather elegantly, and the algorithm certainly instantiated the crucial facet of collision behaviour, the overhead of communication between the different objects made this harder to observe and understand.

A group of students easily built models for the ball-wall situation, and then attempted to model collisions between two balls. The students had not seen the classes of collision as shown in Table 1, but rather had been encouraged to experiment with the velocities of the balls to create different types of collisions and note the outcomes. After modelling the different scenarios, an 11 year old girl, Anne, reflected on what she had found, in explaining to the whole group:

“If something is already moving, and you push it, it’s going to keep moving. But if something is not moving, and you push it, it’ll just start moving. And if something is very solid, and doesn’t move, and you push it, you’ll actually bounce off it.”

What is noteworthy about Anne’s description is that she was spontaneously starting to define different types of collisions arising from her experience with testing the models: her first sentence corresponds to collision class D, her second to class A, and her third to class F.

When we reflected on this first design sequence, we realised that although the fixed-object system was easy to model and students could bring strong intuitions to the situation as to ‘what should happen’, the sequence as a whole did not well suit the objective of students seeking a simple model that fitted *all* the cases, that is the ball-wall *and* all those relating to the two movable objects. We therefore decided that our learning aims would be better met if all collision classes were treated as 1D collisions between movable objects with the fixed-object system considered as a *special case* of the two-movable-objects system, in which one object had infinite mass. From this perspective, it

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

made more sense to position this special case at the end of a sequence of activities that moved from equal masses (classes A-D) to different masses (class E), and ultimately treated one of the masses as tending to infinity to match the ball-wall situation (class F).

Starting the sequence with carts of equal mass

Thus the main change between the first and second iterations of the activity sequence was to start with the equal mass cases (classes A to D), which meant we had to face the challenge of finding a simple ToonTalk design. After much experimentation, we decided we had to simplify the modelling process still further by *providing* a design solution, as shown in Figure 4. Thus rather than expecting students to write ToonTalk programs from scratch, we provided them with the key building blocks for their models. Their challenge was to program the behaviour of the carts in the different cases. In Figure 4, cart1 and cart2 are shown (top right) with the two *sensor* types crucial for collisions shown in the boxes below them: *Collide* (two on the left) which indicate whether the object is colliding or not; and *Right Speed* (two on the right), which not only indicate the values of the right speed but also are remote controls for these variables. These four sensors are referred to as the robot's *input box*. An untrained robot is shown to the left.

[INSERT FIGURE 4 HERE]

Figure 4 needs some elaboration. Training and testing a robot for a given collision situation involves a number of steps. The first step in robot training normally consists of deciding what to put in the robot's input box: many students found this quite a challenge (it requires a considerable thought experiment to predict what the robot will "need"), and

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

so we chose to provide them with a solution. The student begins programming the behaviour of the robot by picking up the input box with her *hand* (shown at the bottom) and dropping it on top of the robot to enter its *thought bubble*. The challenge for students was to determine how to program the robot using these sensors/remote controls, in order for the behaviour of the carts to match their predictions after they collided.

Once inside the thought bubble, the student literally *shows* the robot what to do (constructs a program) by using it to pick up, point to and modify the values of the sensors. Once training is finished, the student escapes from the thought bubble. An important feature of the ToonTalk language is that once a robot has been trained and given its input box to operate on, it will run through its actions for as long as the input box matches the conditions under which it was trained (there is a simple mechanism to *generalise* robots from their initial training conditions). A robot can be inspected by running it *on the floor*, in which case it will run slowly and demonstrate its trained actions through animation. However, a robot and its input box can also be sent off in a *truck* to another *house* where it will do its processing (sending a *truck* to build a *house* is the ToonTalk metaphor for spawning a sub-process: in this case the robot will run quickly as it does not need to display the animation, so that the collision behaviour of the carts would happen in real time).

After training their robot then, the student would typically send it away in a truck to test their model under different collision conditions. They would set the speeds of the two carts to values that represented the collision class under consideration, and then observe the resulting behaviour of the carts when they collided, as controlled by the programmed

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

robot running in another house. If they found that their model did not behave as expected, they could debug their model, and inspect their actual robot by going to the other house and watching its actions as it ran "on the floor" of the house.

While investigating each collision class in the activity sequence, students went through a cycle of prediction, observation of the phenomenon, modelling in ToonTalk, model testing and consideration of the limitations of any current model's scope of application: to assist the reader in visualising the modelling phase of student activity, we have created a 'webreport'⁵. At the beginning of each cycle, students were given a written task which showed a diagram of the pre-collision situation and asked for qualitative and quantitative predictions about what would happen to the cart velocities after collision (see the example in Figure 5). Students were then shown a video of a real-world collision between two carts that satisfied the initial conditions of the class. The video served as an initial motivation for the modelling activity that followed⁶. This led into a teacher-initiated

⁵ See (http://www.weblabs.org.uk/wlplone/Members/gordon/my_reports/Report.2005-02-17.1716), which includes three video clips for collision class A. The first clip shows the collision phenomena used as motivation, the second clip shows a student model being tested, and the third shows the actual workings of the model (i.e. the actions of the programmed robot).

⁶ We are aware of the danger that some students might view the video of the collision events as the phenomena rather than experimenting with an actual physical setup themselves. Nevertheless, the possibility of repeated 'ideal' collisions, together with our focus on generality rather than physics per se, convinced us that this was adequate as motivation for the modelling phase, and as a target behaviour used for comparison with a given model.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

group discussion in which students' written predictions were compared with their video observations. After consensus was reached about what had been seen the students articulated what they needed to represent in their models, and then proceeded to the modelling phase (episodes from which are described in detail later).

[INSERT FIGURE 5 HERE]

Part of the modelling process included students' debugging and validating their models in terms of conformity to the goals they had in mind: did the (virtual) carts behave as predicted, with the behaviours (quasi-laws of motion) they had been given? In other words, students gave the two carts appropriate relative speeds by setting sensor values, and then checked whether the resulting collision behaviour matched their expectations of whether their robots behaved as they intended, and the extent to which the resulting programmed collisions corresponded to the 'real-world' video clips. When they were happy with their model, students posted it with an accompanying description as a report on the WebReports site (as shown in Figure 2). In some instances, students downloaded and tested each others' models and posted comments and modifications on the site. Students were thus encouraged to consider different models of the phenomenon.

The sequence started with consideration of collision class A. When students began to explore the next class of collision, B, (which was again initially motivated by a video), they were provoked to consider the limitations of their initial model, and began to wonder whether their model was a general solution to represent all types of 1-D collision. The activity sequence was therefore cyclical, with students encountering each cycle for a

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

given collision class before considering the next class, with the aim that they would eventually appreciate the benefit of constructing a general model that would work for all classes, rather than a different model for each class. This cycle of investigation of each collision class is illustrated schematically in Figure 6.

[INSERT FIGURE 6 HERE]

Examples of student activity in the modelling phase

We now present some glimpses of students' activity as they worked through the second version of the activity sequence. On the basis of the previous iterations and consideration of the demands of the necessary programming and algebraic knowledge, as summarised earlier, we predicted that students would attempt three different types of models for the collision classes A to D, *transfer*, *reflect*, and *velocity swapping* (as shown in Table 2), with the first two models being subsets of *velocity swapping* that work for particular cases. The *transfer* model is one in which cart1 'gives' its velocity to cart2, and cart1 becomes stationary. We thought that this model would be attempted for collision class A, because students would say that cart1 was *transferring* its velocity to cart2, in a single direction swap in which cart1's velocity is set to 0, rather than its zero velocity being thought of as transferred. We predicted that the *reflect* model in which both carts change direction (sign of velocity) on collision, would be attempted for class B, because it seems more straightforward to say that the carts are both reversing direction (e.g. like a ball bouncing off a wall) than to recognise that they are in fact swapping their velocities. The *velocity swapping* model, in which cart1 and cart2 swap their velocities, works for all

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

four classes A-D, and abides by the laws of conservation of energy and conservation of momentum. We predicted that students would replace their more limited *transfer* and *reflect* models with the more generally applicable *velocity swapping* model, when investigating classes C and D.

On analysing the students' activities as they worked through the sequence, we found that our predictions were largely correct. Students created *transfer* models for class A and *reflect* models for class B, without being instructed to do so. Not only is the ToonTalk programming required for these two models relatively straightforward, but the results look reasonable when compared to what would be expected and what was seen in the initial motivating videos. However, the *transfer* and *reflect* models only give rise to behaviour that looks correct for classes A and B, and students tended to create *velocity swapping* models when considering collision classes C and D, in line with our predictions.

[INSERT TABLE 2 HERE]

We now present some snapshots of student activity while modelling the different types of collision, starting with class A and moving to class D.

Snapshot of modelling collision class A: Specific values and limitations of the transfer model

The first modelling approach to collision class A, adopted by all the students in London, was to train robots to set a *particular* value to the post-collision speeds that satisfied the

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

relationships they wanted to be true. For example, when building a *transfer* model, students would set up the input so that Speed1 was say 5, but instead of copying and assigning Speed1 to Speed2, so that Speed2 was *always* the same as Speed1, they would take a 5 from the toolbox and add it to Speed2, or simply type 5 on the sensor. As an intervention, a researcher would typically suggest trying a different speed (say 20) for the speed of cart1, and the students would see how cart2 always moves at speed 5 after collision, instead of at the incoming speed of cart1. Yet most students would fix this by training another robot to set the speed of cart2 to 20 after collision. Thus students had difficulty in viewing the pre-collision speed of cart1 as a *variable* as opposed to a *value*. In some ways perhaps this should not be surprising – none of the students were expert programmers. Even if students had wanted to copy and assign a sensor value they may not yet have possessed the technical programming expertise to do so easily during their modelling⁷. More interestingly, the idea that one value is functionally related to another (even trivially) is not straightforward, and notoriously hard for students to express algebraically.

A further example of this phenomenon is evident in the work of the students in Cyprus. The students were set the task of examining one another's models for collision class A. One student, Bedros, had posted his ToonTalk model in a webreport and another student, Cosmo, downloaded it for inspection. In this situation, after collision, cart1 should become stationary and cart2 should move with the velocity that cart1 had prior to collision, as illustrated in Figure 7.

⁷ To do this involves using the ToonTalk *magic wand* to copy, and the '=' key to assign.

[INSERT FIGURE 7 HERE]

Cosmo tested the model with different values for the speed of cart1, and found the behaviour to be reasonable. One of the researchers then suggested that Cosmo try pressing ‘-’ on the speed sensor of cart2 after collision, which changed the direction of movement of cart2 so it moved back to collide with cart 1 again. Cosmo tried this and was surprised to find an unexpected behaviour – that cart1 and cart2 ‘stuck together’ and both continued moving to the left, as shown in Figure 8. In fact, Bedros had programmed his model so that when a collision occurs, cart1’s speed was *added* to cart2’s, and then cart1’s speed was set to zero – it only worked when cart1 was moving before colliding. Therefore, when cart2 hit cart1, cart2’s speed was unaltered (had zero added to it) and cart1’s speed was also unaltered (was set to zero). But the robot was also trained to ‘uncollide’ cart1 from cart2, which is a way of ensuring in ToonTalk that the robot will only run once when the balls collide. The result was that cart1 was ‘pushed along’ by cart2 and it appeared they were ‘stuck together’.

[INSERT FIGURE 8 HERE]

Cosmo quickly realised that something was wrong. He said:

“It ought to transfer its velocity to cart1, am I right?”

He was encouraged to write a comment on Bedros’ webreport, and wrote:

“When cart1 travels from left to the right, it’s velocity is transferred to cart2. Can you explain me what is going to happen when cart2 travels from right to the left?”

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

At this point, Bedros and Cosmo had a face-to-face discussion about the problem. Then Bedros answered the question posted by Cosmo, writing:

“The carts won’t behave as we would expect, because they are not trained to travel from right to the left.”

A third student joined and explained why the model must be ‘wrong’. He explained that if the trucks were on a table, and you were watching them from one side (cart1 moves right into cart2), you could walk around to the other side and the directions would be reversed (cart1 moves left into cart2). Obviously the behaviour should be the same when viewed from a different position, but it was not in the ToonTalk model – a very insightful use of symmetry! A discussion about how the model could be rectified then ensued.

Snapshot of modelling collision class B: limitations of the reflect model

In the *reflect* model both carts reverse their direction on collision, and this gives the correct behaviour for collision class B. Two students in London, Rhona and Yvonne, had built a *reflect* model and were encouraged to test it out under the conditions of collision class C, where the carts are moving toward each other with different speeds. In this case, the *reflect* model might appear to work, although its ‘fit to reality’ would depend on the difference in magnitude of the two velocities. If the magnitudes of cart velocities are relatively close, when each cart reverses direction the collision behaviour would appear reasonable. If the magnitudes of the cart velocities are very different, the behaviour would start to look unrealistic, although the carts would still appear to ‘bounce off’ one another. Some students still judged this an acceptable model for class C. However, when

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

the *reflect* model was tested under conditions of collision class D, the case where one cart ‘catches up’ with the other, the resulting behaviour of the carts looked very odd and obviously incorrect, as after collision both carts would start to move backwards. This is shown in Figure 9: when cart1 catches up with and collides with cart2, they both reverse their directions.

[INSERT FIGURE 9 HERE]

This situation was encountered by two students, Rhona and Yvonne, who had built a *reflect* model for classes B and C and then tested it under class D conditions. They immediately rejected their model. They then started to work out what their model should do, as Rhona explained:

“So we started trying to figure out, what would actually happen in real life. So we sort of decided, that in real life, not both of the balls would actually go backwards.

This is going to give its speed, and boost this one.”

Interestingly, Rhona and Yvonne validated their model against their intuitions from the real world, and decided that it did not represent reality accurately. Their reasoning for this appeared to rely on the idea of ‘same’ speed being ‘transferred’ from the faster cart to the slower one – an interesting idea that clearly related to conservation.

Comparing alternative perspectives on modelling collision class C

When students started to consider collision class C, two distinct theories emerged in group discussion. One is illustrated by Sophie, who described the situation in terms of ‘swapping of velocities’, in line with the *velocity swapping* model that she went on to

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

program. Her justification for the swapping model appeared to rely on an intuition of *conservation*.

Researcher: So what about afterwards – if this is 50 and this is -25 , what can we say afterwards the speeds would be?

Sophie: That one would be 50 and that one would be -25 .

Researcher: Why not -30 ?

Sophie: Because it has to keep the same speed of the other one, cart.

By contrast, two other students, Natasha and Chris, described the situation in terms of cart1 *giving* or *transferring* some of its speed to cart2.

Chris: Cart1, when he hits cart2 he will go slower, the speed is going to cart2 so cart2 gets faster.

Researcher: Cart1 hits cart2...

Chris: Yeah.

Researcher: And then its going to go slower because its given some of its speed to cart2 [Chris nods]. I think that's slightly different. I think we have three different models here if only I could get them in my head. What do you think [to Natasha]? Why don't you just try and summarise it for me?

Natasha: Because cart1 is going at a faster speed when it hits cart2 it sort of gives, sort of pushes that one to go faster.

Researcher: Right, pushes that one to go faster...

Natasha: Yeah, then cart1 will go slower.

Researcher: I think that's rather like what Chris said.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

This second type of description bears resemblance to the ‘momentum flow’ conception of mechanics discussed by diSessa (1980). This argues that a legitimate way to think about force is in terms of the flow of momentum from one place to another – in this case momentum can be said to be flowing from cart1 to cart2 during collision. Unfortunately these two students did not follow up on trying to build this type of model in ToonTalk: it is not straightforward to program such a model, and there are conceptual problems to face. How does one determine how ‘much’ speed to transfer from one cart to another? Is it a proportion of the speed, or a constant amount? Students could have started by building a model that transferred a constant amount of speed (adding, say, 20 to one cart, subtracting 20 from the other) which would only work for a specific case. However, this does not really capture the essence of *transferring* speed. One ‘correct’ way to model it would perhaps be to take the difference between the cart speeds, add the difference to the speed of the slower cart, and subtract the difference from the speed of the faster cart. This requires testing to see which cart is slower, which would require a ‘team’ of ToonTalk robots. Also, careful consideration of the sign of the speeds (i.e. direction of velocity) needs to be taken into account with this procedure. Once implemented correctly, this model would, of course, have exactly the same outcome as velocity swapping. However, despite articulating a ‘transferring’ model, the students built a *velocity swapping* model, since this is much simpler to program, and gives the same outcome.

A note on a collaborative dimension through WebReports

We had initially hoped to foster significant cross-site collaboration between the student groups in London and Nicosia through the WebReports system, but this did not occur due to a number of organisational and technical difficulties. There were however some

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

instances of within-site collaboration which are illustrative of the potential of this way of collaborating at a distance, and which are guiding us in the final year of the WebLabs project. In one case, for example, two London students received webreport comments six weeks after finishing activities on 1D collisions. These small comments motivated them to resume their work (at the expense of the current topic), in order to answer the questions posed. The students became highly engaged in their responses, to the extent that they spent significant time debugging and reposting their models. It is also noteworthy that they were able to recall their work from six weeks earlier, perhaps because they could recreate the context easily by reading their webreports and particularly, by opening and inspecting the models that were stored on the site.

Discussion

From a design perspective, this study raises a number of interesting and non-trivial issues. Although it is true that most of our students came to see that the *velocity swapping* model worked best, and certainly that their descriptions of the phenomena became more ‘rigorous’, we do not claim that this arises solely from our design of the technical system. On the contrary, we are aware of the crucial importance of the teacher/researcher’s role, and the ways in which the collaboration between students helped in generating a classroom discourse that supported the spirit of scientific enquiry that we were trying to encourage. We do not, as yet, have sufficient data on this aspect, although we are attempting to gain some in the forthcoming final year of the study.

One aspect of our design that seems to emerge as important is our relatively successful attempts to encourage students to focus attention – implicitly if not explicitly in

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

discussions – on the ideas of system state and conservation. The imperative of generalisation and consistency is one that characterises science and mathematics, but is all but absent from all other pursuits. So it is hardly surprising that students generally find it difficult to accept that contradictory statements of behaviour cannot adequately describe the physical world, even though they are typical of all other human discourse! Our design decision to focus attention on *simple* systems (1D is, of course, a very special case of collision) that were sufficiently interesting, and amenable to the imposition of a simple model, was – in retrospect and despite several iterations to get it right – supportive of encouraging students to view the phenomenon from a scientific point of view. We saw, implicitly, that the intuition of conservation, for example, emerged in some of the student discussions, in the form of both swapping velocities and transferring some velocity (both of which imply that total velocity is conserved). The concept of system states was taken up by the students – they were comfortable with evaluating the pre- and post-collision states of the system while ignoring the interactions of the collision itself. This was mirrored by the programming implementations in which a single robot ran only once on collision, giving rise to a single change in state of the system.

The most common limitation of students' work while modelling was that they tended to produce specific rather than general solutions. This meant their solutions would only work correctly for one set or a restricted class of 'input' speeds. Over time and with help however, students did see the value of general-case rather than specific-case models. It is interesting that the common bugs were to do with making specific solutions that did not generalize.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Students encountered a number of difficulties when programming with ToonTalk in the modelling phase. This is to be expected of course, and learning to overcome various programming challenges can be considered an important by-product of the activities. The type of difficulties can however be divided into two categories; those related to the learning of some programming technique or concept, and those where the metaphor of ToonTalk is not well suited to the programming task at hand. The former is an acceptable difficulty, whereas we wanted to minimise instances of the latter. Examples of both type occurred when students needed to swap the values of sensors when building the *velocity swapping* model. When swapping the value of two variables in any programming language, a temporary storage variable is required. An analogy can be made to the swapping of two different liquids contained in glasses; first transfer one liquid into an additional glass, then transfer the other liquid into the empty first glass, and finally transfer the liquid from the additional glass into the second glass. Students had difficulty when first encountering the need to swap the values, but managed to implement a swapping procedure with help from instructors. However, there were additional ToonTalk programming complications that arose when implementing the algorithm⁸.

⁸ The sensors need to have their 'data type' converted to values, before swapping them. This is because sensors are more like *pointers* than *variables*, and changing the value of one copy of a sensor changes the value of all other copies. Although it can be argued that this is another example of learning a programming concept (albeit a more advanced one), there is an additional problem in that sensors are not really displayed as pointers when training a robot. For technical reasons, the values of sensors are frozen when entering a robot's thought bubble (training). The result is that sensors do not change value when training the robot, the swapping procedure appears to work during training, and only breaks down when the robot is actually running. This was difficult to explain to students.

Recognising that the *velocity swapping* model is superior to the *transfer* and *reflect* models because of its generality (in the respect that it accounts for the post-collision behaviour in all four classes A-D) is one thing, but there is also a deep question about how the model relates to physical reality. Is it really the case that the carts are swapping velocity with one another? Physicists would probably say not. Rather, their behaviour is the result of the carts obeying the laws of conservation of momentum and conservation of energy, and that these happen to simplify to velocity swapping in these restricted same-mass, 1D cases. Velocity swapping then is more of a consequence than an *explanatory* theory. Indeed, this was the reason that modelling collisions with different-mass balls (classes E and F) were originally included in the classification – to show that the *velocity swapping* model has limited scope and is only applicable to same-mass situations. However, due to time limitations it was not possible to undertake the final two models, although these will be included in the next design experiment.

Next steps in design

We note the importance in our activity design of collaboration both face-to-face, and at a distance via the WebReports system. An example of within-site collaboration was given earlier, in the context of Cypriot students testing one another's models. In fact, we believe that this testing of each others' models should, in the next iteration, be incorporated into the activity sequence specifically as a cross-site activity. This will, of course, raise practical problems, as the collaborating groups would need to be engaged in the same stage of the activity sequences at similar times.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

In the work detailed above, we did not report on collision classes E and F, and these need to be built into the next iteration in order to complete our design objectives. As we have already discussed class F, we now briefly outline some design challenges for class E collisions. We would have liked students to be able to build a model for different mass collisions themselves (class E), similar to their model building and testing in classes A-D and class F. However, for different mass cases there is no easy way to simplify the formula (i.e. equations 3 and 4 for the post-collision velocities), or build a model that does something other than instantiate the algebra. Our proposed solution was to implement the model as a pre-built tool with which students could experiment.

Figure 10 shows a prototype we have constructed, which we term the *conservation of momentum* model. The algebra is effectively hidden by sending the robot off in a truck to another house⁹, so that students see the results of colliding the carts but not the workings of the model. The mass of the carts is calculated based on their size (surface area of rectangle), and can be directly manipulated using the ToonTalk *bike pump*.

[INSERT FIGURE 10 HERE]

We have three important reasons for including the different masses case in the activity sequence. First, we want students to realise that velocity swapping is not in fact a general solution for all types of collision, and is limited in scope to 1D same-mass collisions.

⁹ This is one of the means by which ToonTalk allows abstraction of processes: the inner workings of the new process are carried out in another 'house' so that it is not necessary to see how its results are calculated.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Second, we want students to test the *conservation of momentum* model and discover that it works not only for different masses (collisions E and F) but also for carts of the same-mass (collisions A-D), and is thus a more general model than the *velocity swapping* one. Third, we want students to experiment with the model we provide, and gain some qualitative insights into the relationships between velocity and mass. We do not expect students to discover or formally learn the conservation of momentum law, but rather come to understand that, for example, a faster or more massive cart has a greater effect on a collision than a slower or less massive one.

We have yet to try these last activities *in situ* with students, but informal piloting suggests that when the post-collision velocities of the two carts are simply derived from the application of the conservation of momentum and energy laws, rather little understanding is gained as to *why* the carts move at these velocities after collision. The mathematics provides the answer, but rather little predictive power, as the algebra is too complex. However, this informal piloting revealed that the use of the *conservation of momentum* tool *did* furnish some intuitions. For example, its use in modelling revealed a sort of conservation of relative speeds (specifically that $v_1 - v_2 = v_2' - v_1'$). This can be proved algebraically, but is not immediately obvious from the equations, and was a surprise to researchers despite the fact that they had built the model themselves. Another discovery was that if the mass of cart1 was made very small compared to that of cart2, and cart1 was set to move into a stationary cart2, then the behaviour of the model approximated rather well to the special case class F (cart1 bounces off with almost opposite velocity, cart2 moves very slowly). This is because cart2's mass is approaching infinity relative to cart1, so it is approximating a ball-wall situation. The addition of this

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

activity means that the design process had gone a full circle, as class E approached the ball-wall situation at the end of the sequence rather than at the start, as we had intended in the first iteration of design.

Concluding remarks

In this paper, our focus has been on epistemology and design, and we have yet to achieve a position in which we can make any realistic claims concerning students' learning, although we have provided some snapshots that illustrate the potential of the approach for learning. Our approach is predicated on three basic ideas: that students build models for themselves; that we attempt to provide tools at the right grain size simultaneously to facilitate model-building, and to afford examination of the structure of tools and models we provide; and that the possibility of reflecting and commenting on others' ideas is enhanced by the possibility of critiquing and rebuilding actual models.

This kind of approach is unlikely to lead directly to learning of conventional scientific curricula in the short term, but it might, we believe, form a substrate on which future learning can be facilitated. From this perspective, the focus on the fundamental concepts of states, systems and conservation makes sense: but it makes it more difficult to assess what learning may be taking place during (and after) engaging with the activities. We can, however, be fairly sure that these ideas are seldom explicitly encountered in the traditional algebraic approach, where there is a seemingly inevitable emphasis on the manipulation of the symbols at the expense of meaning.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

This assumption is, we have argued, clearly the case with respect to the laws of conservation. They are expressed beautifully and concisely in the conservation equations, the power of algebraic representation at its best. This is what the equations are designed to do: to sum up the invariants of the behaviour of physical systems in an equal sign! Yet while this is an undisputed advantage for representing and predicting already-understood laws of motion, it is difficult for the novice to unpack the meanings of conservation as a system invariant, rather than being able to use the specific instances of conservation represented by the equations.

From a pedagogical perspective, this, in fact, the essence of the constructionist vision: that by building entities for oneself, one might come to understand deep structural relations that are generally only implicit. We did, as we have reported, gather some modest evidence that through our design sequence, students actively engaged in the process of modelling began to seek out invariant laws and specifically came to see the value of general-case models, rather than making models that only worked in specific cases. This latter finding should not be underestimated: a substantial element of research in mathematics education indicates just how difficult it is for students to see that rules of all kinds need to apply across cases (and that it is the delineation of a set of cases that leads to definitions). Students routinely apply different (and incorrect) procedures to situations without questioning whether it really *can* be true that – say – multiplying or adding can be used arbitrarily to achieve a given outcome!

Our design experiment also pointed to the challenges inherent in our approach; the time and effort needed for iterative design, not least because we cannot automatically assume

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

that learning to articulate relationships in ToonTalk (or in any other computational system) is easier or even more expressive than it is in algebra. Each representational system, however, affords different ways to say things and – no less important – different things to say. In this respect, we can view our work as a contribution to the development of complementary (rather than alternative) infrastructures for expressing mathematical or scientific relationships.

A final point concerns the question of design and grain size. It is difficult simultaneously to provide students with building blocks that are sufficiently powerful to create models, yet sufficiently flexible and transparent to encourage students to interrogate their inner workings. In fact, for much of the modelling process we erred in the former direction, and provided key building blocks rather than expecting students to build their own from scratch. Perhaps we cannot expect to get this balance correct in general – students will differ tremendously in their own priorities and interests, even assuming that they remain focused on the task at hand! Nevertheless, exploring the question of grain size, and attempting to strike the right balance between functionality (the tools do a useful job) and transparency (the tools can be inspected, manipulated and modified) remains a key priority for future iterations.

References

Cobb, P., Confrey, J., diSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational Researcher*, 32, 1, 9-13.

Constantinou, C. P. (1999) The Cocoa Microworld as an Environment for Developing Modelling Skills in Physical Science. *International Education and Life-Long Learning*, 9 (2), 201-213.

de Jong, T., Martin, E., Zamarro, J.-M., Esquembre, F., Swaak, J., & van Joolingen, W. R. (1999). The integration of computer simulation and learning support: An example from the physics domain of collisions. *Journal of Research in Science Teaching*, 36 (5), 597-615.

diSessa, A. A. (1980). Momentum flow as an alternative perspective in elementary mechanics. DSRE working papers, MIT.

diSessa, A. A. (1988). Knowledge in pieces. In G. Forman & P. B. Pufall (Eds.), *Constructivism in the computer age* (pp.49-70). Hillsdale, NJ: Erlbaum.

George, E. A., Broadstock, M. J., Vazquez, & Abad J. (2000). Learning Energy, Momentum, and Conservation Concepts with Computer Support in and Undergraduate Physics Laboratory. In B. Fishman & S. O'Conner-Divelbiss (Eds.), *Fourth International Conference of the Learning Sciences* (pp. 2-3). Mahwah, NJ: Erlbaum.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Grimellini-Tomasini, N., Pecori-Balandi, B., Pacca, J.L.A., & Villani, A. (1993).
Understanding Conservation Laws in Mechanics: Students' Conceptual Change in
Learning about Collisions. *Science Education*, 77 (2), 169-189.

Hoyles, C., Morgan, C., & Woodhouse, G. (1999). *Rethinking the Mathematics
Curriculum*, London, UK, Falmer Press Ltd.

Hoyles, C. & Noss, R. (in press) Playing with rules: the collaborative construction of an
adventure game. In: Blatchford J. (Ed), *Experimental School Environments*. Cambridge
University Press.

Kahn, K. (1996). ToonTalk - An Animated Programming Environment for Children.
Journal of Visual Languages and Computing, 7 (2), 197-217.

Kahn, K. (1999). From Prolog to Zelda to ToonTalk. *Proceedings of the International
Conference on Logic Programming 1999*.

Lawson, R.A., & McDermott, L.C. (1987). Student Understanding of the work-energy
and impulse-momentum theorems. *American Journal of Physics* 55 (9), 811-817.

Louca, L., & Constantinou, C. (In Press). The use of computer-based microworlds for
developing modeling skills in physical science: An example from light. *International
Journal of Science Education*.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Noss, R. (2001) For a Learnable Mathematics in the Digital Culture. *Educational Studies in Mathematics*, 48, 21-46.

Noss, R., Hoyles, C., Gurtner, J-L., Adamson, R. and Lowe, S. (2002) Face-to-face and online collaboration: appreciating rules and adding complexity. *International Journal of Continuing Engineering Education and Lifelong Learning* 12, 5/6, 521- 539

Mor, Y., Tholander, J., & Holmberg, J. (In Press). Designing for cross-cultural web-based knowledge building. *Proceedings of CSCL '05: The Tenth International Conference on Computer Support for Collaborative Learning*.

Sherin, B. L. (2001). A comparison of programming languages and algebraic notation as the expressive languages for physics. *International Journal of Computers for Mathematical Learning*, 6, 1-61.

Sherin, B. L., diSessa, A., & Hammer, D. (1993). Dynaturtle revisited: Learning physics through collaborative design of a computer model. *Interactive Learning Environments*, 3 (2), 91-118.

Sherr, R.E., Shaffer, P. S., & Vokos, S. (2001). Student understanding of time in special relativity: Simultaneity and reference frames. *Phys. Educ. Res., Am. J. Phys. Suppl.*, 69 (7), S24.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Smith, J.P., diSessa, A.A., & Roschelle, J. (1994). Reconceiving Misconceptions: A Constructivist Analysis of Knowledge in Transition. *Journal of the Learning Sciences*, 3, 115-163.

Figures

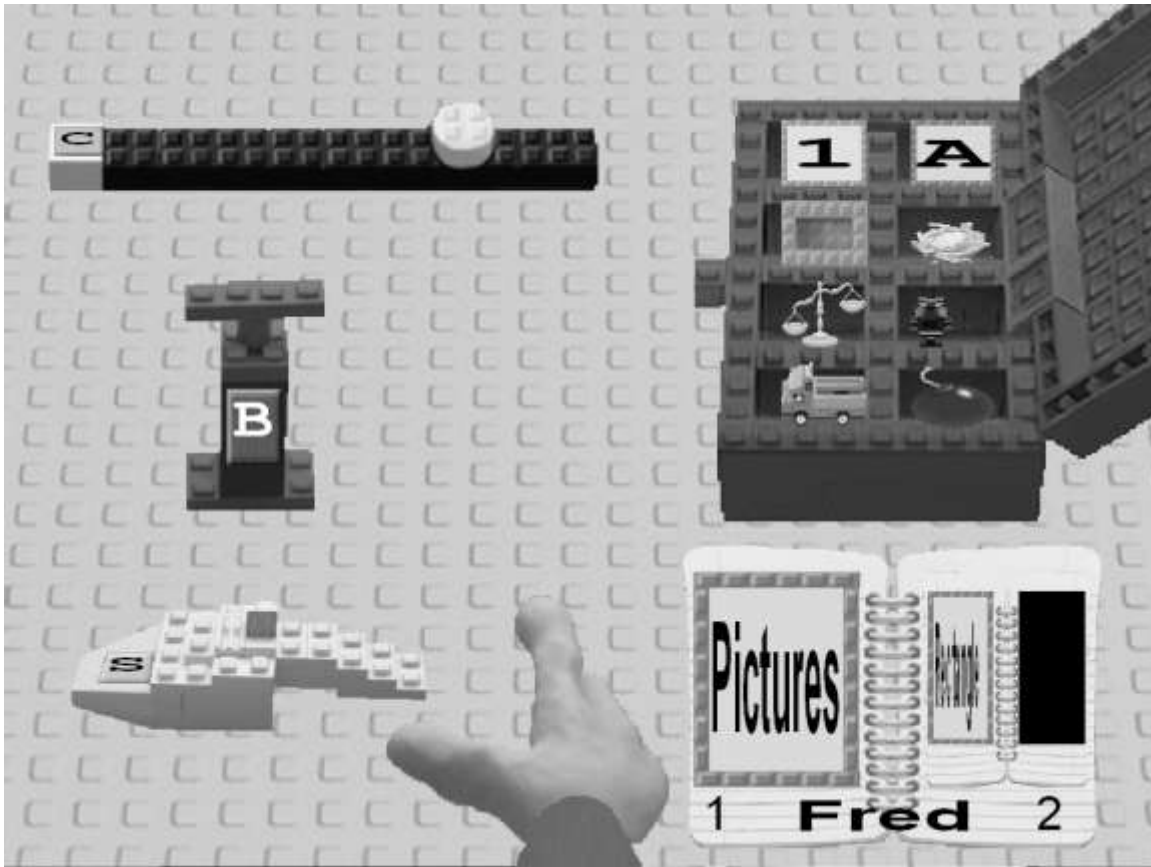


Figure 1

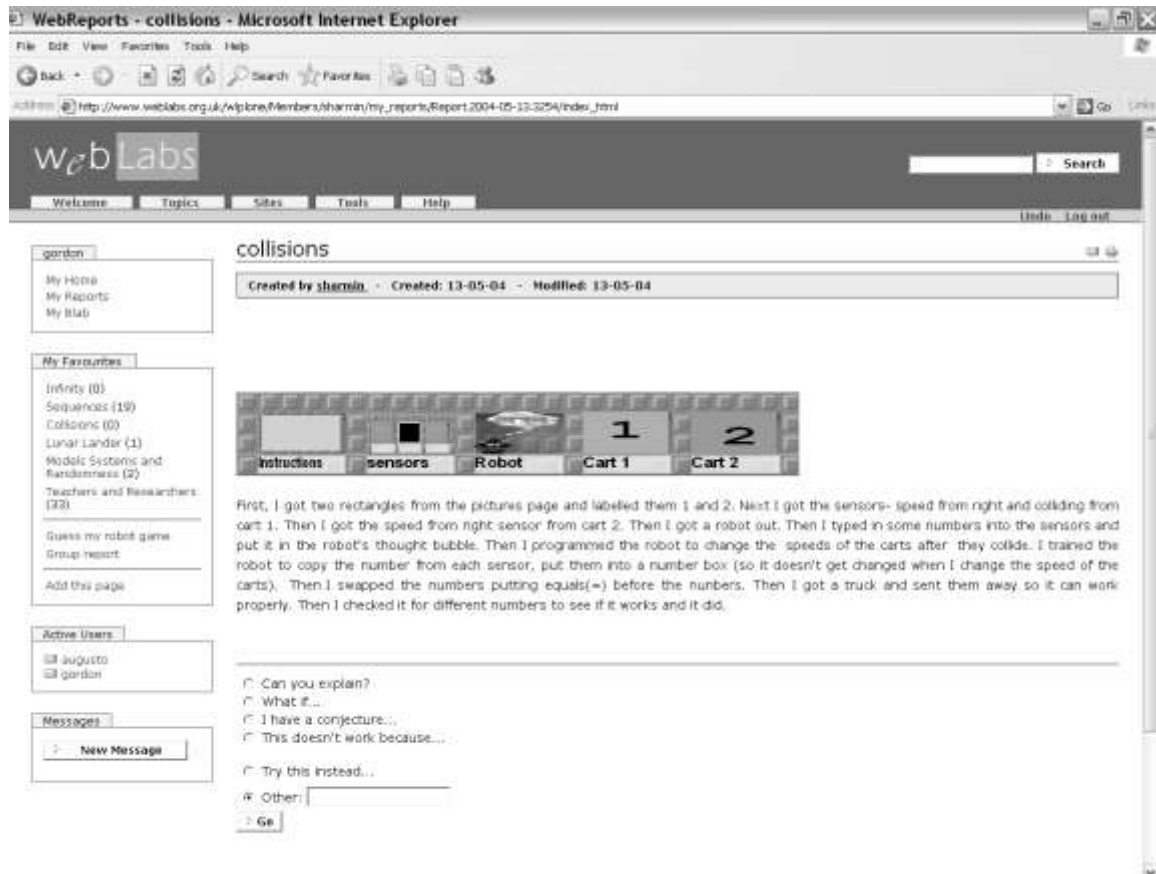


Figure 2

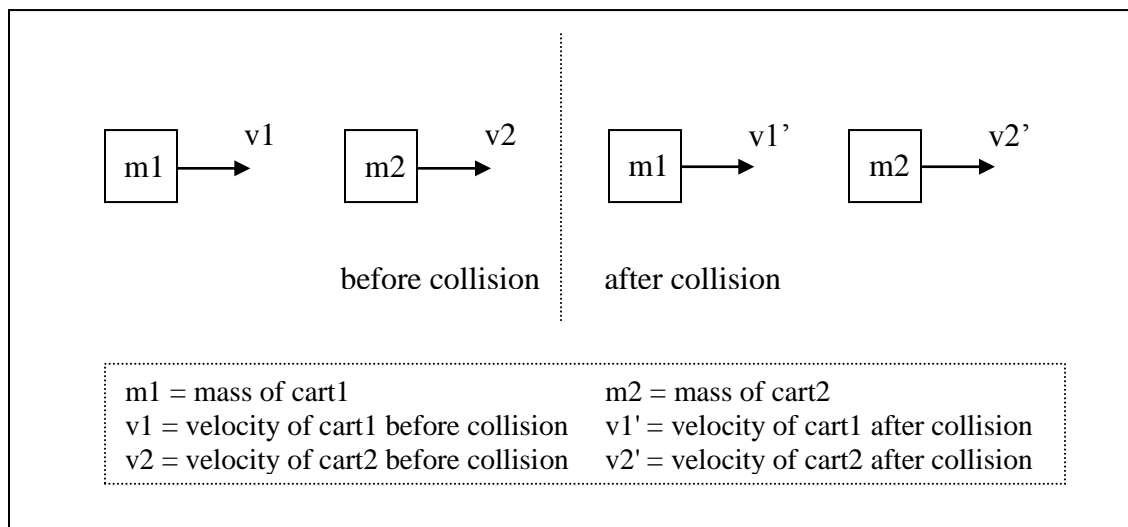


Figure 3

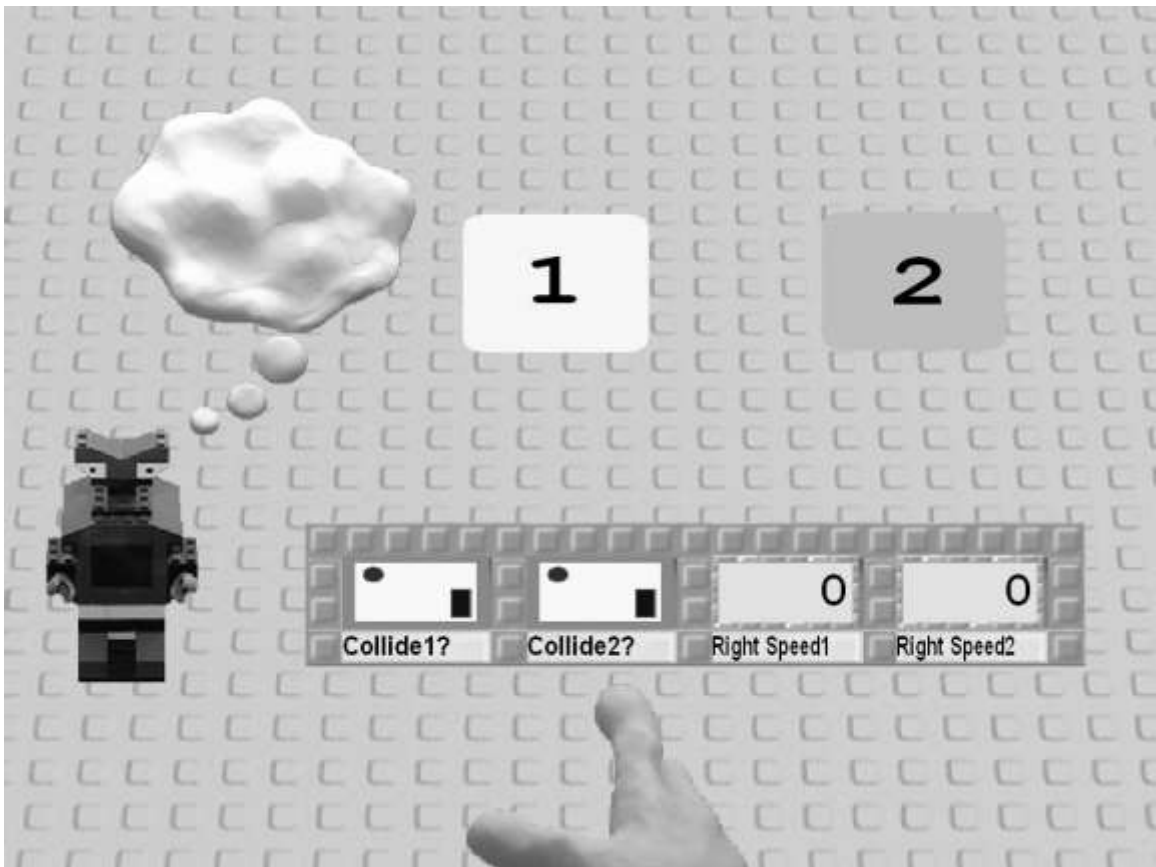



Figure 4

Shown below are two identical carts on a frictionless surface. Cart 1 moves to the right with a speed of 50 units, while cart 2 is stationary.



1. What do you think will happen, when the carts collide?

2a. Which of the following numbers represents the velocity of cart 1, after it collides with cart 2? (circle the right answer)

-100 -50 -25 0 25 50 100

2b. Explain why you chose this number.

3a. Which of the following numbers represents the velocity of cart 2, after it collides with cart 1?

-100 -50 -25 0 25 50 100

3b. Explain why you chose this number.

Figure 5

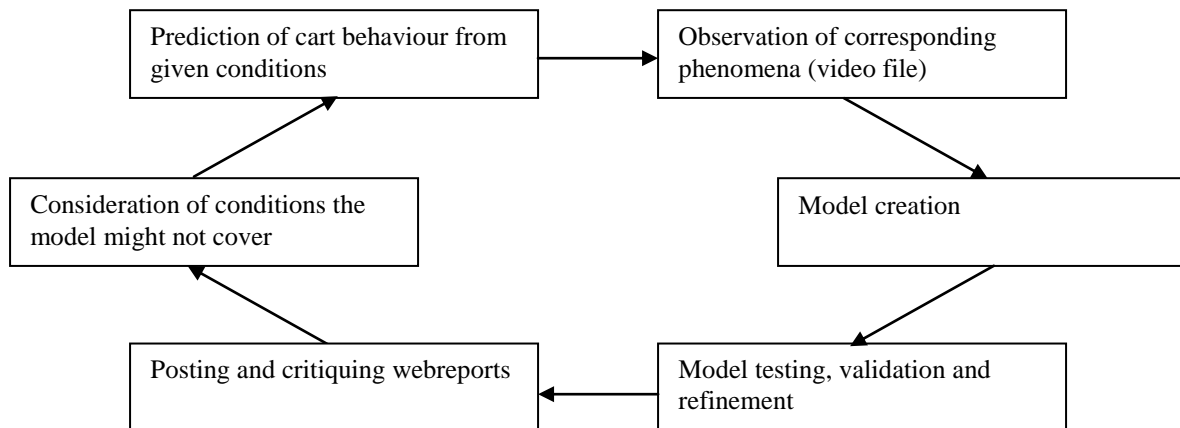


Figure 6

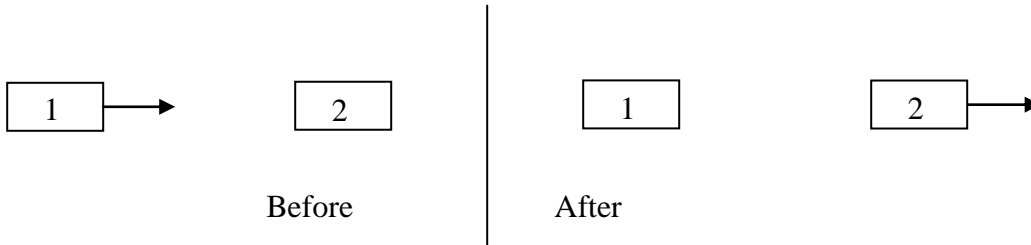


Figure 7

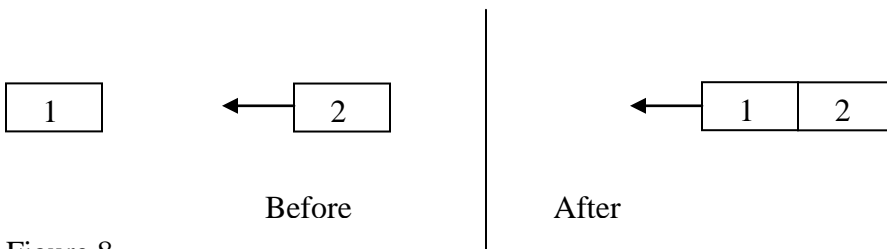


Figure 8

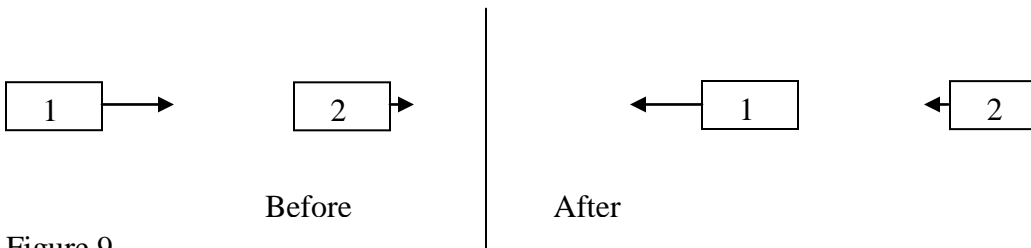


Figure 9



Figure 10

Figure legends

Figure 1. A snapshot of the ToonTalk programming environment. The user can pick up (with the virtual hand), and use the magic wand, the bicycle pump and the vacuum cleaner. The user also has a main notebook which contains built-in pictures and sounds, mathematical functions, and controls for various properties of the programming environment, along with a 'toolbox' of primitive objects containing number pads, text pads, boxes, bird-nest pairs, scales, robots, trucks and bombs.

Figure 2. A snapshot of the WebLabs WebReports site. This page shows a student's embedded model with an accompanying description of how they programmed it in ToonTalk. A navigation menu is shown at the top and quick links on the left of the page. The functionality to add comments can be seen at the bottom. The user selects from a pre-defined list of comment titles or writes their own, before proceeding to the wysiwyg editor.

Figure 3. Diagram of before and after conditions for 1D collision between carts.

Figure 4. The design of the model for programming the behaviour of two colliding carts. The input box containing the *Collide* and *Right Speed* sensors/remote controls is shown to the right of an untrained robot. The pictures representing cart1 and cart2 are shown at top right.

In *Journal of Computer Assisted Learning*, (2005) 21, pp143-158

Figure 5. An example of the written probes that students completed for each collision class, before viewing the corresponding videos of the situation.

Figure 6. The repeating cycle of phases in the activity sequence: 1D collisions between carts.

Figure 7. Normal collision class A behaviour.

Figure 8. The unexpected behaviour in collision class A when cart2 hits cart1.

Figure 9. The obviously incorrect behaviour of the *reflect* model when tested with collision class D.

Figure 10. The *conservation of momentum* tool developed for horizontal collisions between carts of differing mass.

Tables

Table 1. Pre- and post-collision diagrams of the six different classes of collision.









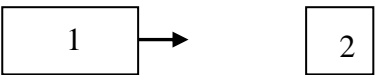


Collision Class	Pre-collision	Post-collision	Cart Masses
A: One cart stationary			Same mass
B: Equal and opposite			
C: Opposite direction, different magnitude			
D: Same direction, different magnitude			
E: Different masses		Depends on relative masses and velocities	Different masses
F: One cart infinite mass			One cart with infinite mass

Table 2. Possible types of model for collisions A-D.

Collision	Diagram	Transfer	Reflect	Velocity Swapping
A	○-> ○	Yes	No	Yes
B	○-> <-○	No	Yes	Yes
C	○--> <-○	No	Possibly: depends on relative values	Yes
D	○--> ○->	No	No	Yes