Pearce, J. M., & Livett, M. K. (2001). *MotionWorkshop: Tracking Motion in an On-line Environment.* Paper presented at the Apple University Consortium Conference, Townsville.

MotionWorkshop: tracking motion in an on-line environment

Jon M Pearce
Department of Information Systems
University of Melbourne
Victoria, 3010
(613) 8344 9248
Fax: 9349 4596
jonmp@unimelb.edu.au

Michelle K Livett
School of Physics
University of Melbourne
Victoria, 3010
(613) 8344 8071
mklivett@unimelb.edu.au

About the authors

Jon Pearce is a Senior Lecturer in the Department of Information Systems where he has research interests in Web-based learning, multimedia and interactivity. He previously lectured physics within the University and has had a long involvement in physics education and designing interactive physics learning environments. Michelle Livett is a Senior Lecturer within the School of Physics where she coordinates the first year physics programme. She carries out research into laser-tweezing and also has been involved in designing physics learning environments for many years.

Abstract

This paper describes an AUDF funded project to produce a Java applet to help with the teaching of physics. The applet, called *MotionWorkshop*, allows students to track the motion of an object in a video clip, display the resulting data in a spreadsheet and manipulate graphs representing the motion. The spreadsheet supports the generation of a numerical model of the motion that can then be compared with the actual video data. Innovative features of the program let the student manipulate the spreadsheet in a manner that facilitates exploration of the model. The use of *QuickTime* with embedded *Flash* tracks is described as a convenient way of adding comments and help information to the program.

Background to the project

This project has a long history–it began as a 1996 National Teaching Development Grant project called *Real-World Physics* (for an early description see Pearce and Livett, 1997), which aimed to help teach first-year physics students about physics in the real world. One component of that project was to create a Java applet to allow students to track the motion of an object in a video clip. This applet was to be embedded in a web environment comprising learning exercises, video-clips of real-world events, and video interviews of the main players in the events.

The Java applet, called *MotionWorkshop*, became the focus of much further design and development work. Other stand-alone software was developing at this time to carry out video analysis, and some of these packages still are available (see, for example, *VideoPoint* (www.lsw.com/videopoint), *Measurement in Motion* (www.learn.motion.com/lim/mim/Measurement1.html) and *World in Motion* (members.aol.com/raacc/wim.html). However, these could not be run via the Web, and generally did not have the sophistication of modelling and analysis tools that we required.

In 1996 Java had only recently been released and we suffered through the various vagaries of the early versions. Although video was central to our needs, the early Java did not support *QuickTime*. To overcome this deficiency, we had to export our movie clips as sequences of GIF images and hence emulate 'movies'. As *QuickTime for Java* became a reality, we applied for funding from the Apple University Development Fund to implement *QuickTime* in the project and to add various other features. We received a major AUDF grant in 2000 and, after a slow start, have been making progress in redeveloping aspects of the program.

Overview of the project

The aim of *MotionWorkshop* is to enable students to analyse the motion of real-world events captured on video, and use their knowledge of physics to describe and analyse the events. *MotionWorkshop* allows students to analyse events such as the motion of a long-jumper or the swing of a golf club. The event must first be videoed and saved as a *QuickTime* movie. Students use the applet to track the motion of a point or points on the moving object(s) by clicking, frame by frame, on the video image on the screen. The resulting position data are recorded into an on-line spreadsheet that automatically calculates values of velocity and acceleration. From the spreadsheet students can plot graphs and enter formulas to carry out numerical analysis of the motion, or model new motions.

The spreadsheet has been designed to simplify the generation of data using numerical modelling techniques. This allows students to create a mathematical model for a motion they are studying and see the model's results overlaid on the analysed video data. This form of modelling is a powerful method of "completing the loop" in helping students to understand motion; it enables students to set up their own models of the causes of motion, and compare the outcomes with reality.

How is *MotionWorkshop* used?

Analyse the video

Typical use of the software starts with a student choosing a video clip; several have been produced specifically for this project, such as falling balloons, bouncing balls, juggling clubs and sprinters. The student would then choose an object within the video whose motion she wants to analyse. For example, in analysing the motion of a falling balloon, she would choose a point on the balloon and click on it frame-by-frame to enter its (x,y) position coordinate data into the spreadsheet. The coordinate axes on the video can be moved, rotated, or even changed to a polar coordinate system, and the data in the spreadsheet will update automatically.

Having entered the data she could choose to have the software display other kinematic values such as velocity or acceleration. These would be listed in new columns in the spreadsheet. She might choose to calibrate the data so that screen pixels are converted into units of metres or centimetres. It might be of interest for her to see other quantities derived form the displayed ones, for example, kinetic energy. To do this she would enter a formula into a column of the spreadsheet using syntax similar to *Excel's* 'row-column' syntax. It has been a deliberate design decision to let the system calculate and display only 'kinematic' quantities only (position, velocity and acceleration); the student must define the relationships for other 'derived' quantities (force, energy, momentum, etc.) for herself.

Having set up the spreadsheet in this way she could display one or two graphs of any columns against any other column. In the balloon example she might choose to view velocity versus time and acceleration versus time to see how the balloon quickly reaches a terminal speed as the acceleration drops to zero. These graphs are easily scaled or shifted by dragging the axes.

Model the motion

Having been able to observe and analyse the nature of this motion, the student can construct a numerical model by entering the relevant equations of motion into new columns in the spreadsheet. For our balloon example, she would want to explore how air resistance affects the acceleration of the balloon. She would do this by entering formulae relating the forces acting on the balloon to the motion of the balloon; these would appear in new columns in the spreadsheet. The data she constructs in this way can be overlaid as position data on the original video motion to test its validity. *MotionWorkshop* provides easy ways for her to change variables in her model and immediately see the effects. "What if" type questions are asked in accompanying worksheets whereby she is motivated to consider variations on the motion and test her hypotheses by calling up a related movie clip and again comparing this real motion to her analysis. Further information on this style of modelling is discussed below.

Learning: the importance of the activity

We have been engaged in investigating the use of *MotionWorkshop* to enhance students' learning about the relationship between forces and motion (Pearce et al, 1998; Rodrigues et al, 2001). Although the primary purpose of *MotionWorkshop* is not to replace hands-on laboratory experience, our initial investigation focussed on its use as an alternative to a parallel lab activity. This research indicated that *MotionWorkshop*-based learning activities can improve student understanding. However, it emphasised that good learning outcomes are not simply due to this software but are the result of carefully crafted learning activities in which the software use is embedded.

Innovations: thinking differently

During the early stages of developing *MotionWorkshop* some asked why we did not use an existing spreadsheet, such as Excel, and export data to it for analysis. There were two reasons for this. This first was that it was important for us to have tight integration between the video component, the spreadsheet component and the graph component. The students can see the position of an object, e.g. a juggled ball, on a frame of the video, and simultaneously see its corresponding points on a graph as well the value in a cell in the spreadsheet. The links between these representations of the motion help her to construct her understanding of the motion. She could also drag an object in one component and see the corresponding change in the other. Manipulating these multiple representations of objects in this fashion is an important pedagogical feature as students can directly observe how changing one representation affects another.

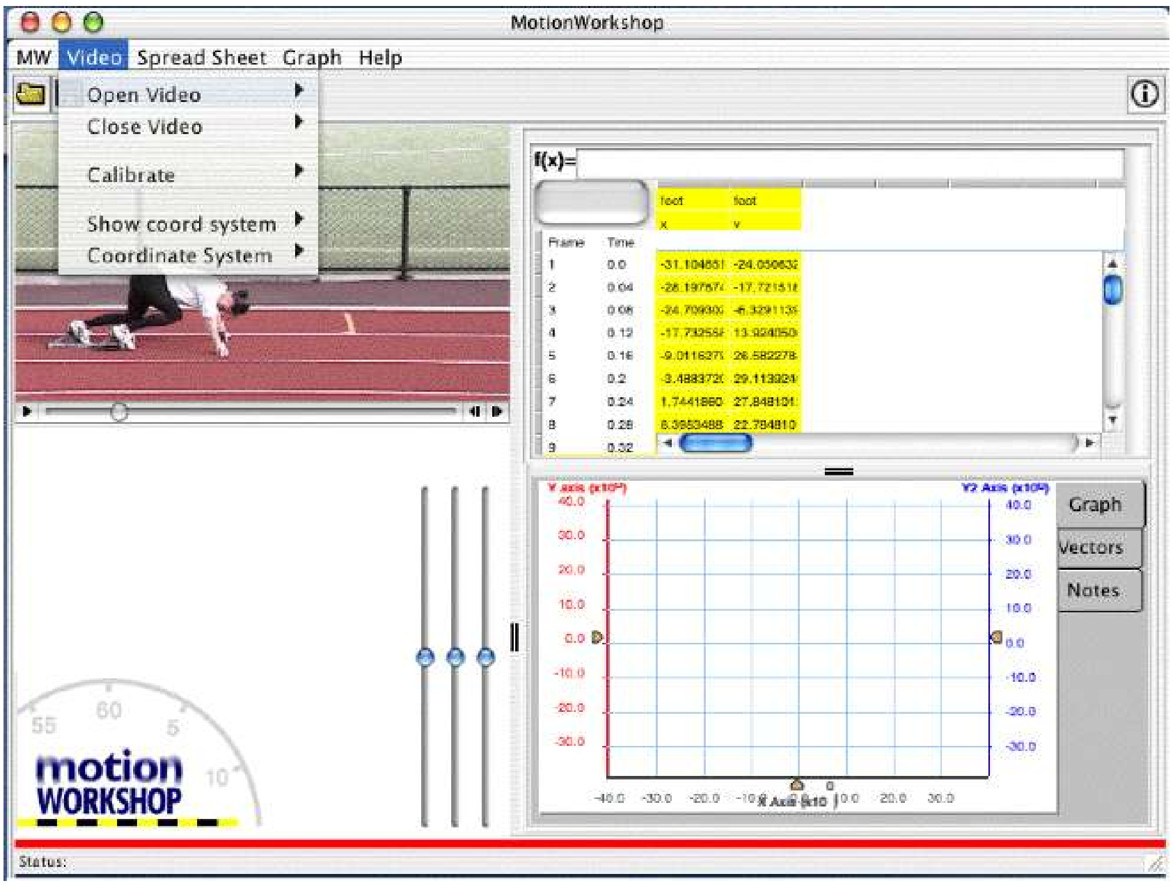
The second reason relates to not being satisfied with the way that existing spreadsheets allow the user to model physical events. But first, a few words about the nature of the modelling that students do. The modelling is referred to as “numerical modelling”. It is a powerful technique when you want to model a motion of something for which you cannot write an explicit formula: maybe it is too complex, or maybe an exact analytical solution does not exist at all. Instead, you regard the motion as comprising hundreds of tiny ‘steps’ in time, where each ‘step’ is calculated from the previous step by applying the underlying laws of motion. Hence the whole motion of the object is simulated by carrying out many calculations, each representing changes during very small time intervals (maybe one hundredth of a second, for example). With each calculation a new value is entered into a spreadsheet column. An example of a motion well suited to this form of analysis is a falling object—say a balloon—for which the exact mathematical solution is complex, but the physical laws to be applied at each small step in time are relatively straightforward. One advantage of this form of analysis is that constructing the mathematical model relies very heavily on understanding basic laws of physics at each time interval, rather simply applying mathematical techniques; the student stays focussed on the physics rather than the mathematics.

Hence, given the nature of the modelling that students would do with the spreadsheet, we required an improved interface between student and spreadsheet. There were two issues here: firstly the spreadsheet would have a large number of rows due to the very short time intervals required to accurately model a few seconds of motion; and secondly, students need an easy way to ‘tweak’ variables in the spreadsheet and compare the results with the real data as captured by the video clip.

The problem of a large spreadsheet is compounded by the fact that data from a video would usually be at 1/25 second intervals (frame rate for PAL video). The data generated numerically by the modelling process could be at 1/100 second interval, or even more frequent (as determined by the desired accuracy of the model). This results in many rows of the spreadsheet that nobody really wants to see, and which do not line up with any ‘real’ data from the video. We needed a spreadsheet in which the rows could be collapsed to display only, say, every 10th or 20th row, yet the calculations still be valid in the hidden rows. *MotionWorkshop* allows the student to adjust the number of rows displayed to be fewer than, or more than, the original number determined by the video clip.

The second issue, the ‘tweaking variable’ requirement, is an attempt to give students a better way to manipulate their models. To make good use of such a spreadsheet and use it to explore a model in comparison with real video data, we needed an easy way for students to change the value of variables and see an immediate update to the spreadsheet and accompanying graphs. For example, when modelling the motion of a falling balloon, the student must consider three forces: the weight of the balloon, drag (friction) and buoyancy. Values for the weight of the balloon and buoyancy are easy to obtain, but the coefficient for drag is not so easy to find out. However, students are able to make an initial guess and link this spreadsheet constant to a slider in order to be able to vary its value easily and see the displays update in real time. In this way, the student’s model could be tweaked to obtain a better match between model and observation.

Figure 1 shows an early screen shot of *MotionWorkshop* set up to analyse the motion of a sprinter. Effort is currently being put into implementing Java's "*Swing*" features. This gives the appropriate GUI look-and-feel for whichever platform is being used (*MacOS*, *OS X*, *Windows*, *Linux*). The figure shows the look-and-feel under *OS X:* transparent menus; *Aqua* sliders; etc. Using *Swing* offers many other benefits apart from look-and-feel. It simplifies programming considerably by providing objects such as sliders, tables, drop-down menus, as well as features such as tool tips. The programmer has less code to write and produces a more consistent product. The disadvantage is that the initial download of the *Swing* library is quite large, but this is now included in most standard Java installations (including *OS X,* but not *OS 9*).



		foot	foot
		x	y
Frame	Time		
1	0.0	-31.104651	-24.050632
2	0.04	-28.197674	-17.721518
3	0.08	-24.709302	-6.3291139
4	0.12	-17.732558	13.924050
5	0.16	-9.0116279	26.582278
6	0.2	-3.4883720	29.113924
7	0.24	1.7441860	27.848101
8	0.28	6.3953488	22.784810
9	0.32		

Figure 1. Early MotionWorkshop *screen showing* Aqua *look-and-feel.*

The two features mentioned above make the *MotionWorkshop* spreadsheet very different from the likes of *Excel*. It allows a different approach to numerical modelling in which one can see the important data within a model and easily make changes to values in the model, watching graphs update in real time. There is now a tight integration between video, spreadsheet and graph providing a very powerful modelling environment.

Integrating *QuickTime* and *Flash*

One requirement for this project, which became apparent only after much of the design work was already complete, was the ability to display information about each movie the students chose to load. Students need to view data such as background information, frame rates, scale, time duration, etc. Requirements such as these that come *after* the Java programmer has finished present a problem for academics who have no Java programming experience. There

were also other areas in which we needed the flexibility to be able to change things without employing further ‘hard’ programming expertise: adding help support, ‘how-to’ instructions, links to related web sites, etc.
A solution to this problem was to use *Flash* tracks embedded within each movie-clip (see Figure 2). This menu-like feature at the start of a movie can also be used to present some general ‘what to do with this movie’ information to give students some further support. It can also link to web pages to give more detailed help. The advantage of these is that they can be added to the movies at any time and require no re-coding of the applet.

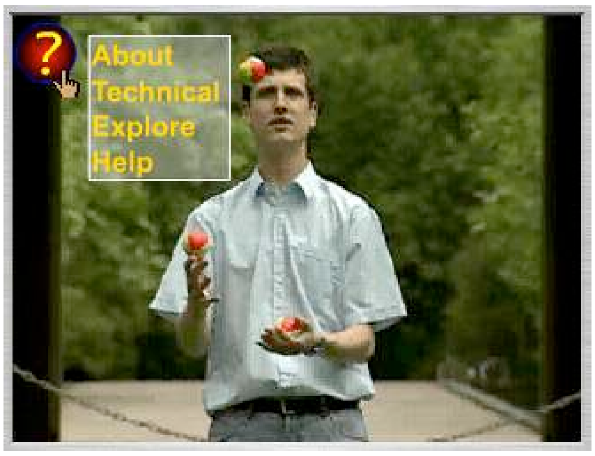


Figure 2. QuickTime *movie showing a* Flash *drop-down menu*

This use of *Flash* in a *QuickTime* movie can be extended by having a blank movie placed on the screen (by the programmer) as a background graphic that we can use at a later date to add instructions, acknowledgements, help information, tutorials, etc. It provides us with a very flexible way of adding changes to the applet later using the more common skills of a multimedia programmer rather than the expensive and rare talents of a Java programmer.

Conclusion

MotionWorkshop has undergone a long development process, evolving in response to several significant external technology changes. From its start as a small Java component embedded in a web-based learning environment, it has grown into an application in its own right. It is still evolving and nearing the stage where it can be released for others to use.

References

Pearce, J. M, Livett, M. K. and Rodrigues S. 1998. *Development and use of an on-line video-analysis tool for physics learning*, Apple University Consortium Conference, Melbourne.

Pearce, J. M. and Livett, M. 1997. *Real-World Physics: a Java-based Web Environment for the Study of Physics*, Proceedings of AusWeb97, Brisbane, July.

Rodrigues, S., Pearce, J. M. and Livett, M. 2001. *Using video analysis and dataloggers during practical work in first year physics*. Educational Studies , Volume 27, No. 1.