Journal of Universal Computer Science, vol. 2, no. 12 (1996), 805-828
submitted: 19/6/96, accepted: 19/6/96, appeared: 28/12/96 © *Springer Pub. Co.*

Delivering Hypertext-based Courseware on the World-Wide-Web

David Marshall
(University of Wales, Cardiff, United Kingdom
dave@cs.cf.ac.uk)

Stephen Hurley
(University of Wales, Cardiff, United Kingdom
steve@cm.cf.ac.uk)

Abstract: This paper considers the framework in which World-Wide-Web based online courseware can be developed. This framework is illustrated using courseware developed for parallel computing, C programming, X-Window programming and computer vision.
Key Words: Courseware, World-Wide-Web, online, parallel computing, C programming, computer vision.
Categories: H.5 (Information Interfaces and Presentation), H.1 (Information Systems: Models and Principles), K.3 (Computers and Education), H.4 (Information Systems Applications).

1 Introduction

The use of computers to provided an *integrated environment* for teaching a variety of disciplines has received much attention in recent years. Indeed many frameworks[3] have been developed for such purposes. The material provided by such *courseware* varies greatly from the provision of lecture notes and lecture support material through to integrated and interactive tutorial packages. Until recently courseware has existed as *stand alone* packages, however with the advent of the World Wide Web (WWW)[5] on the Internet and accompanying WWW (hypertext) browsers, such as Netscape and HotJava, the provision of courseware has taken on a whole new dimension.

Many subjects can benefit from the provision of such courseware. Indeed we are probably fortunate that our chosen disciplines lend themselves to such methods. For example many methods that we describe in our courses are interactive and can take on many states depending on the input data. It is difficult and/or time consuming to convey all such possibilities in a lecture (or on static text such as handouts or textbooks). Integrated courseware has an obvious advantage in presenting such material.

2 Developing Courseware

In this section we aim to summarise our approach to developing courseware on the WWW. We begin by highlighting the advantages of using the WWW for courseware. We then broadly describe how the work has been implemented.

Developing courseware is not merely a matter of preparing a series of lectures, linking them together and packaging them as a course. A comprehensive design strategy must also consider how to implement and manage a course, how to evaluate the materials that are used, and how to assess the learners. Therefore a systematic approach to the development of course materials. According to Rowntree [18] the systems approach should incorporate four basic strands:

1. Identify course aims and objectives;
2. Develop necessary learning experiences;
3. Evaluate the effectiveness of learning experiences; and
4. Improve the experiences in the light of evaluation.

In this paper we will concentrate on steps 1 and 2 and show how they affect courseware development.

3 Identify Course Aims and Objectives

One of the initial, intended goals of our courseware was that it can be used to support a variety of courses, perhaps including undergraduate degree programmes in computer science, physics, all branches of engineering, mathematics and electronics, as well as the basis of training courses run by computer service departments. The challenge of designing learning materials for such a diverse group is to make the materials approachable for all classes of user, and yet maintain a high degree of specialism, for example, relevant to the field of parallel computing or computer vision.

Designing such materials confronts many well-established and accepted instructional design principles. The first step in many instructional design models, is to analyse the learners who will use the materials. Analysis of even a subset of potential users, however, would have proved expensive and time-consuming. Therefore a compromise was made by putting effort into ensuring that the material would appeal to a broad audience.

The characteristics of an instructional medium which interacts with the learner are the tasks that might influence the learning process [8]. One concern might be how the choice of hypermedia might affect learning. Advantages [8] of hypermedia include giving the user control over the learning process. However disadvantages [9] have cited the lack of feedback and guidance given.

There can be little doubt that the WWW has become the most successful networked multimedia hypertext based system in recent years. The *HTML* language used in WWW documents is extremely simple and yet powerful to use[6, 7, 10, 11, 16]. These factors highly influenced our choice of hypermedia implementation systems. We believe that in the careful design of implementation we have addressed some of the critiques of multimedia, *e.g.* feedback is provided by parts of our courseware including the automated assessment of exercises.

However several severe restrictions in the current WWW protocol mean that more advanced *hypermedia* systems need developing.

The recently developed Java programming is significant here because it makes the WWW truly interactive by incorporating applications that can be programmed, run live and distributed in a simple, safe and portable manner. Java also provides an extensible method to handle, internally, new data type and protocols. Briefly, one can think of Java as a simplified, safe, and device independent version of C++.

How can this influence Courseware on the WWW?

The innovations provided by such *second generation browsers* provide many interesting possibilities with respect to developing courseware. Applications can now become truly interactive. Also Significant advances in incorporating a full range of media have been made. Distributed hypertext linking over the WWW should also be improved.

4 Develop Necessary Learning Experiences

The materials originally designed at Cardiff were based on lecture notes from existing courses. The use of lecture materials is a logical foundation on which to build a course. On their own, however, lecture notes are insufficient. User activities during learning are more important in determining what is learned than the presentation of instructional material [17]. The aim of evaluating the original lecture notes was to convert them into more effective learning materials. To accomplish this, the initial lecture notes in HTML format were evaluated using models developed from principles of instructional theory. Evaluation was undertaken at an organisational level and instructional level. The organisational level focuses on courseware structure, by means of analysis of users and evaluation of the learning that has taken place. At the instructional level evaluation was concerned with the educational effectiveness of unit content.

4.1 Organisational Unit Design

We initially considered several Instructional Design models and eventually adopted an established instructional design model: the ASSURE Model developed by Heinich, Molinda and Russels[4]:

A Analyse Learners
S State Objectives
S Select Media and Materials
U Utilise Media and Materials

R Require Learner Participation
E Evaluate and Revise

The use of the ASSURE model for initial evaluation allows for the systematic alteration of existing course material (lecture notes, *etc.*) by focusing on learning issues which might not have been addressed in the original lecture notes, such as the potentially diverse characteristics and experiences of users. For example, this led in many cases to the alteration of language to suit a more general audience.

4.2 Instructional Unit Design

The instructional level of evaluation is concerned with increasing the educational potential of each unit. This evaluation framework was adapted from Gagné's sequence of Instructional Events [2], which are based on the hypothesised sequence of internal stages of information processing derived from studies of cognitive processes.

The use of Gagné's events of instruction as an evaluation tool led to further changes in the development of a suitable model and particular changes in courseware content. For example instructional event three, stimulating recall of prerequisite learning, led to the insertion of additional references to other units.

Utilisation of the ASSURE model and Gagné's Instructional Events is not intended to provide a prescriptive design model. Rather, it provides a framework based on sound instructional strategies within which it is possible for individual course designers to develop a dialogue about design strategies. The tools provide a common ground for collaboration.

4.3 Utilise Media and Materials

The following sections contain illustrations of two implementations of hypermedia incorporated into the courseware.

4.3.1 Using Mpeg Movies to Animate Algorithms

The *mpeg* movie format is the most popular storage format for image sequences on the WWW. Most browsers are able to support such a format. Animation of algorithms is clearly a useful learning tool[11]. Illustrations can be compiled off-line and simply stored and played back on request. Our courseware has extensive use of such a facility. Example uses of mpeg movies are given in Section 6.

4.3.2 Using Forms and Scripts to Achieve User Interaction

User interaction in hypermedia environments is often limited to selecting options with a mouse. In such an environment, the learner is merely presented with the information, having few opportunities to interact with the material. The HTML language however provides opportunities to develop additional types of participation. Consider the following example:

Simulated annealing is a non-trivial multiprocess whereby the loads on a processor network may be minimised (usually not optimally). Observation of students has shown that involvement in the implementation of this algorithm improves their comprehension of it. The courseware implements a simulation which allows the user to execute the algorithm on a simple linear processor network. The user is able to see the results of the algorithm by means of a graph which plots the load on the network against the "work" done by the algorithm. To enhance understanding the user can adjust the various parameters which affect the algorithm's performance. The algorithm can be rerun with different parameters, and the new results are plotted on the same graph with the previous results. This allows the user to compare and contrast performances, and understand how the parameters affect the algorithm, and consequently more fully understand how the algorithm works.

The demonstration of simulated annealing is possible because HTML allows links to executable programs and scripts. As long as a suitable HTML document is produced by the program(s) called, the user is unaware of all the "behind-the-scenes" operations that are taking place.

The main feature of HTML which allows user interaction is the HTML form, which enables user input to be passed to the programs which produce the HTML documents. We have also made extensive use of forms and scripts to provide comprehensive search facilities within our courses[10, 11, 16]. This is a popular tool since it provides an easy means to access parts of the course in a similar manner to the index of a book. Example uses of forms will be provided in Sections 6 and 7 which deal with specific implementations of courseware.

4.3.3 Using Java for true algorithm animation and user interaction

The recent innovations provide by the Java language and its ability to integrate runnable applications *live* over the WWW provide many exciting possibilities.

The first version of the courseware[10] was implemented before the advent of HotJava. All the background processing performed in the initial versions *live* was achieved by running (Perl) scripts and C programs with the resulting data and images mapped back to the HTML browser (Section 4.3.2). The second version of some of the courseware has the processing routines rewritten, where appropriate, using Java applets for complete user interaction. This was not a major problem as the routines were available in C and easily modified for the (C++ lookalike) Java language. Examples of the courseware featuring Java *applets* are given in Sections 7 and 8.

5 Overview of Current Implementations

The courseware we have implemented currently integrates the following:

- Course notes, program listings, reading lists, class information *etc.*
- Algorithm animations *e.g.* interactive image processing over the WWW.
- Links to run programs and view program output.
- Exercises and solutions.
- Links to other sources of information on the WWW.

The courseware follows the basic framework laid out above in that major topics are basically treated as chapters of a hypertext book. Sub-topics are sections and subsections of each chapter. Many examples of results are given in the form of image sequences. These are simply hypertext links to images. One other development is that *live* data processing can be performed on the WWW. This has been achieved by spawning programs, Java applets or scripts in manners described previously[10, 11, 16].

6 Parallel Computing Courseware

The material produced was developed with the aim of aiding the teaching of high performance computing throughout the university sector in the UK. The course consists of four *books*:

1. *A Classification Scheme for Parallel Computers.* This book describes Flynn's classification scheme for parallel computers.
2. *Interprocessor Communications.* This book describes two methods by individual processors within a parallel computer can send messages to other processors within the same machine.
3. *Load Balancing and Task Scheduling.* This book describes two methods by which an algorithm written for a parallel computer can achieve high efficiency by ensuring that all the processors are performing equal amounts of work.
4. *Parallel Algorithms.* This book describes the writing of algorithms for parallel computers.

To describe the essential features of the parallel computing courseware we will describe the book on Interprocessor Communications. Initially, after selecting this section from the four books mentioned above the user is presented with the page given in Figure 1.

Figure 1: Interprocessor Communication Front Page

This page (Figure 1) provides the following high-level information to the user:

- The top left area of the page outlines the contents of the book. This provides information not only on the titles of the book's chapters but also provides a means of directly accessing the chapters of interest (by clicking on its title) e.g. clicking on *Shared Memory* takes the user directly to this section in the book. The contents list also indicates the current position of the user in the book.
- In the bottom left part of the page, navigational aids are provided *(i)* to move around the information contained in particular books and *(ii)* to allow movement between different books. The various options are illustrated in Figure 2
- The right part of the page usually contains three subsections:
  - At the top the *Outline* section presents a very general and brief description of the topic under consideration, i.e. in this case interpro-

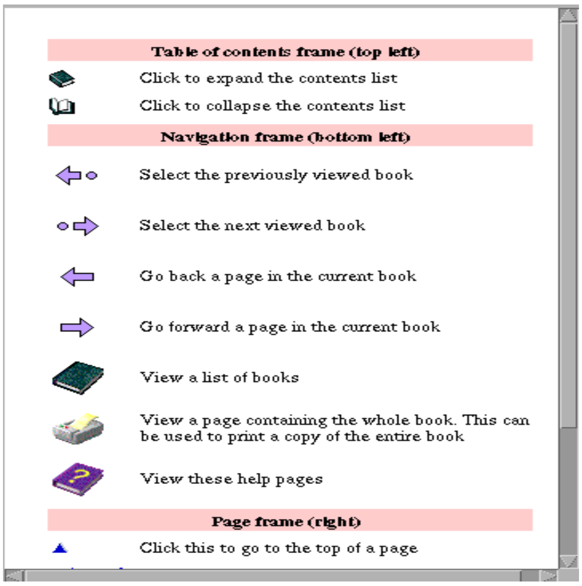

Figure 2: Navigational Options

cessor communication. The purpose of this is to allow the user to decide quickly whether the correct page has been accessed.

– The main part is the *Course* section. This gives a more detailed description of interprocessor communication (or whatever topic is under investigation) and includes links to relevant new topics such as Shared Memory and Interconnection Networks.

  Highlighted keywords in italics (and underlined), are links to background information which the user may need reminding of. For example, clicking on *parallel computing* gives a brief definition of parallel computing and is illustrated in Figure 3.

  Highlighted keywords or phrases that are not in italics are links to detailed information on the concepts under consideration, e.g. selecting *Interconnection Networks* would present the user with detailed information on their description, give examples, show mpeg movies etc. (e.g. Figure 4).

– The *Further Reading* section provides details of reference books (or articles), with page numbers, so that additional information can be obtained on the topic under consideration.

Overall, the courseware has been developed and constructed such that the user (i.e. student) is provided with concise relevant information and is not overwhelmed with too much detail per page. Navigational aids are constructed so

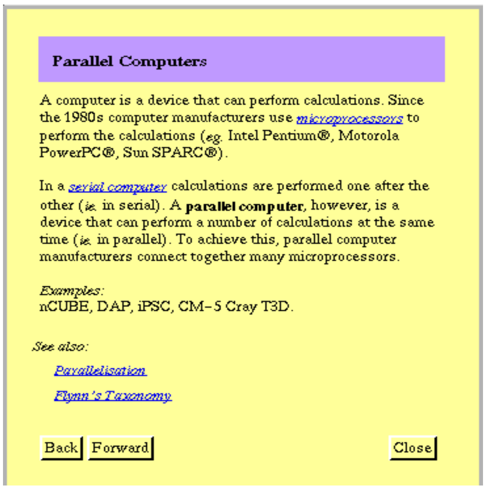

Figure 3: Background Information: Parallel Computing

that the students can search the complete course and individual books relatively easily. The general principles used in developing the parallel computing courseware can be applied to any subject areas.

7 C Programming

The basic design methodology of Section 5 has been extended in the following way for the C programming courseware [1, 11, 13, 14, 15, 16]:

- A form based key word search CGI has been implemented (Figs. 5 and 6).
- At the end of each chapter an exercises section was provided which adhered to the provision of exercises in the lecture course.
- Links for each exercise were provided to the Ceilidh automatic program marking systems ([19]) to provide to the following :
  - Model solutions to exercises.
  - Interface to Ceilidh's automatic program marking module (Figs. 7, 8 and 9).
- Links so that listings of programs can be obtained at relevant points in the notes.
- Interactive Java Applets to illustrate key data structure concepts (*e.g.* Linked Lists and Trees) have been implemented (Figs. 10, 11, 12 and 13).
- Links to alternative notes around the WWW.

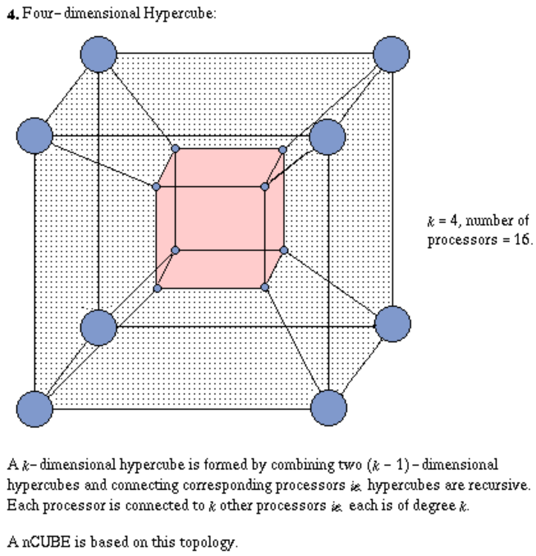

Figure 4: 4D Hypercube Description

- Links to local on-line information about using the workstation, C compilers, UNIX programming and on-line manuals.

In order that familiar information be easily accessible we have implemented a *key word search* facility (Figure 5). This uses the HTML form interface and calls custom written PERL scripts to implement to search of specified sections of the text. The output of a search is a HTML page containing links to documents in which the key was found. Also, within each document, pointers are placed (Figure 6) to indicate the location of the key. This continuing the hypertext text book analogy would correspond to the *index* of a book.

The interface to the Ceilidh Program Assessment system [19] provides are very informative feedback loop for students learning nearly all aspects of C programming. Students submit exercises via a html form (Figs. 7 and 8) and receive a detailed breakdown of their submission (Fig 9).

One recent addition to the C courseware is the use of interactive Java applets to illustrate key data structure or algorithm design. Two examples are Linked Lists and Binary Tree sorts. In both applets the user can assemble an initial structure (Figs. 10 and 12) following this the user can interact to add, delete, *etc.* values. At each level of interaction the applet animates the process and highlights portions of pseudo-code displayed on the screen (Figs. 11 and 13).

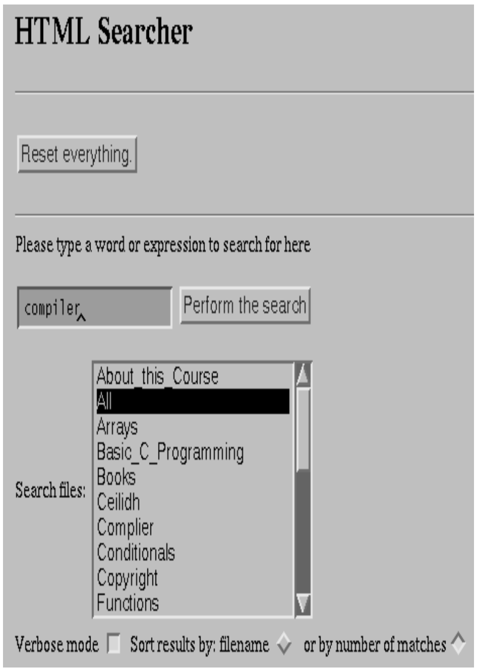

Figure 5: Form Based Key Word Search

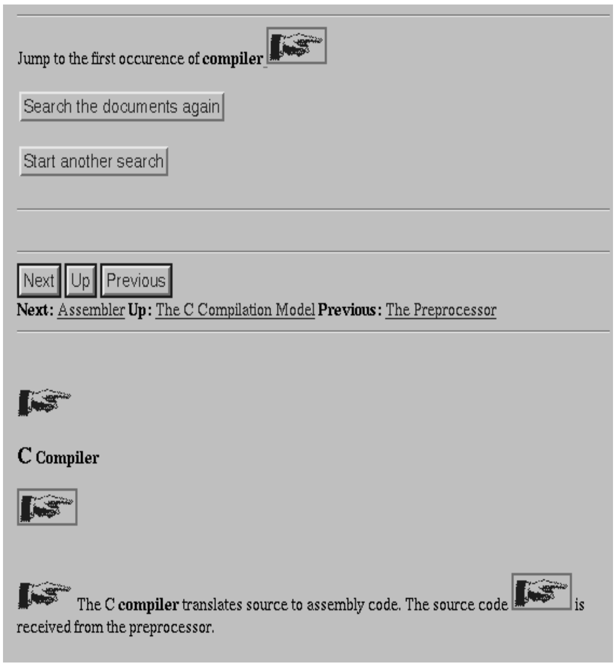

Figure 6: Result of Key Word Search

Figure 7: Program Assessment Interface

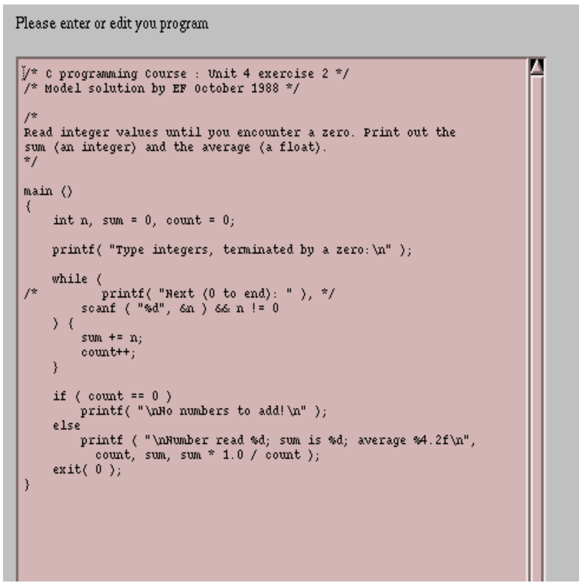

Figure 8: Submitting an Exercise Interface

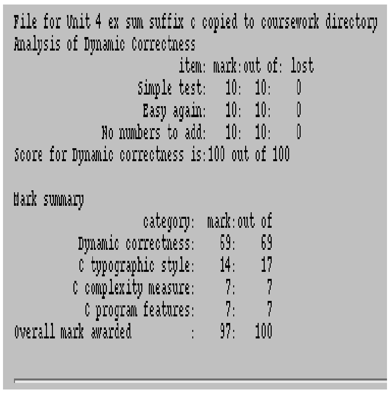

Figure 9: Program Assessment Feedback

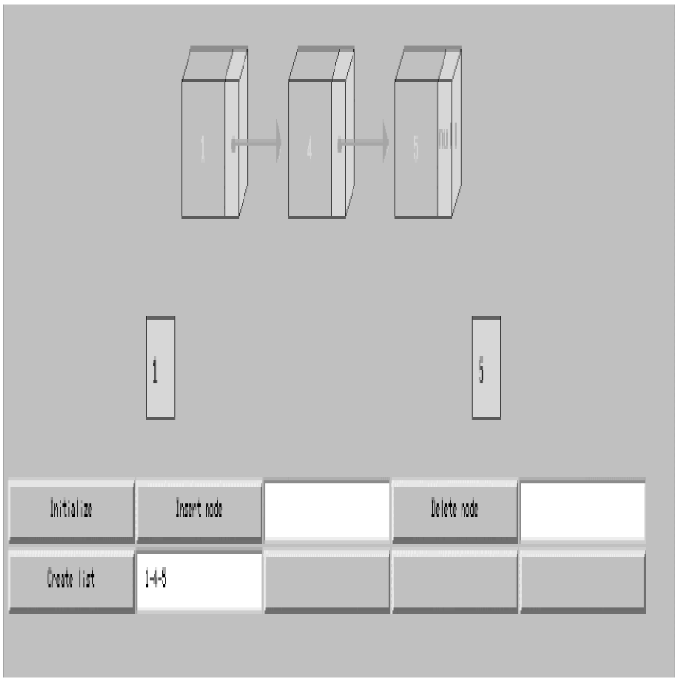

Figure 10: Linked List Java Applet

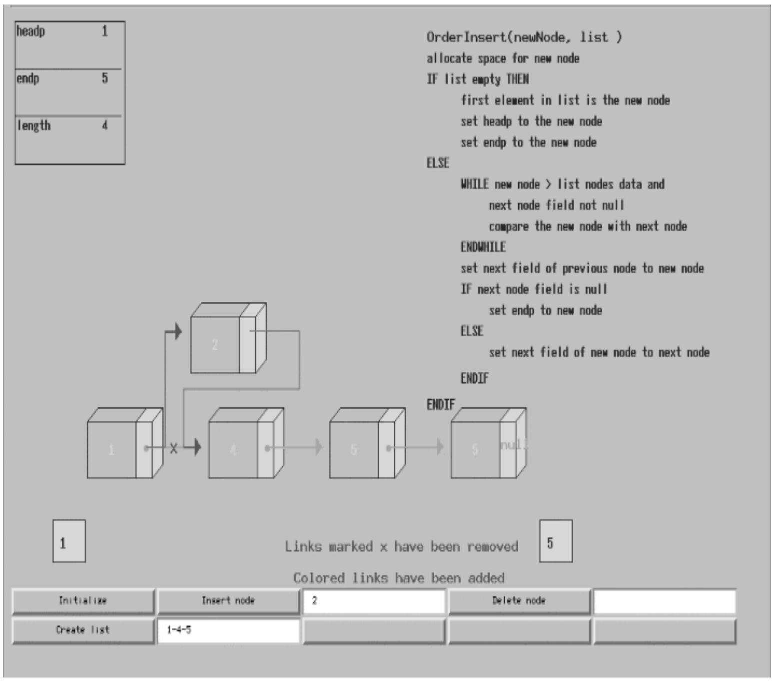

Figure 11: Java Linked List Insertion

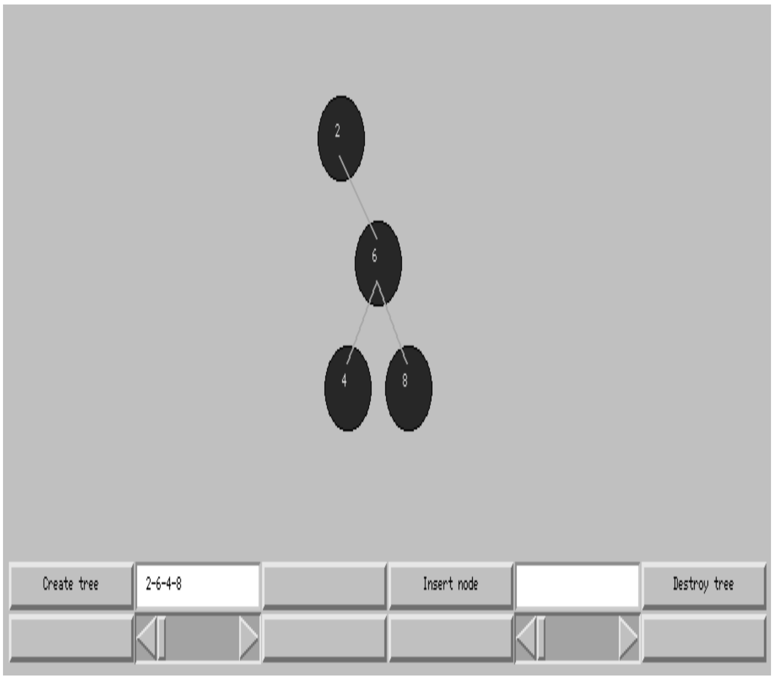

Figure 12: Java Binary Tree Sort Applet

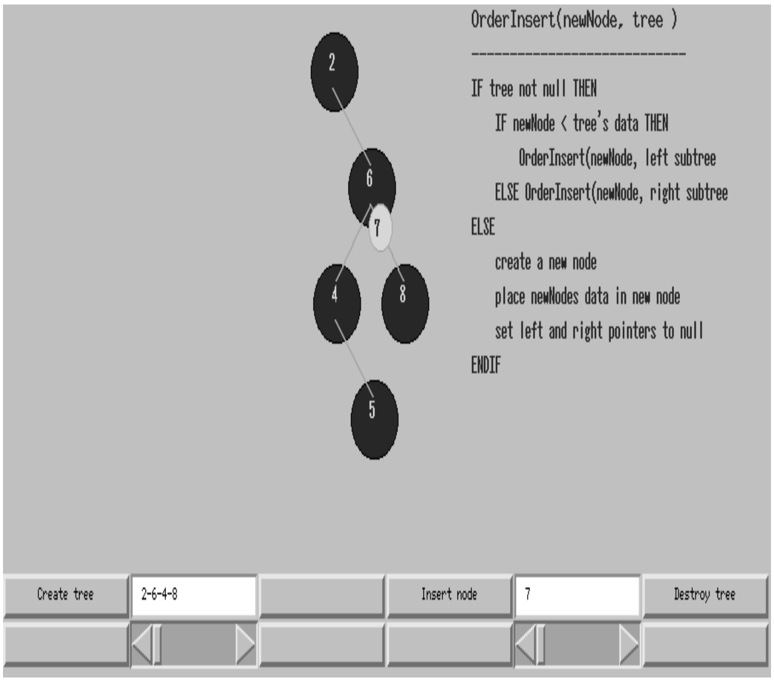

Figure 13: Java Binary Tree Insertion

8 Other Courseware

In this section we highlight important aspects of two other courses we have developed – X Windows and Computer Vision. The basic design and development of the courses is as has been outlined throughout this paper. However both these disciplines involve a high degree of interaction in understanding key concepts. The interactive facilities provided by various facets of our courseware is clearly useful in giving a clear understanding and exploration of these concepts.

X Windows

The X Windows courseware features a novel Java applet that allows the assembly of X Windows interfaces graphically within the courseware. The user simply submits the WYSWIG appearance of the interface to produce X Window C code that can be downloaded and compiled as actual X Window program code. The X Window *Builder* Interface is illustrated in Figs. 14 and 15.

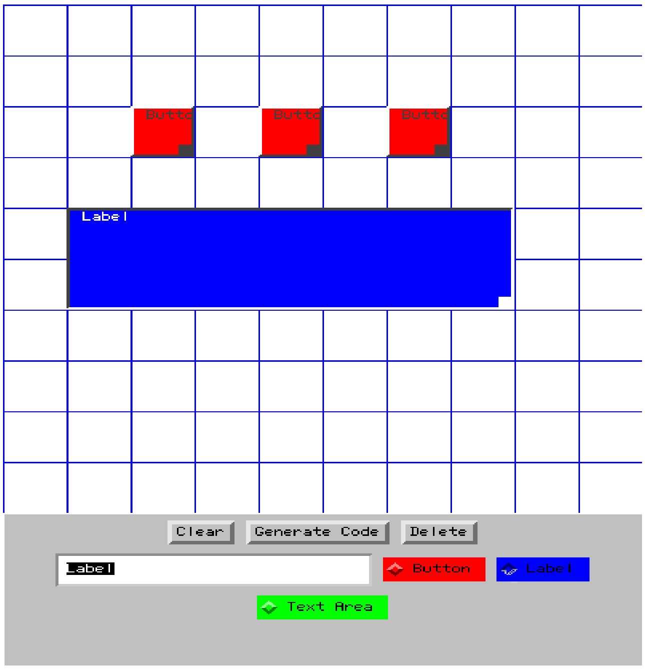

Figure 14: X Window Builder Applet

Computer Vision

Many issues in Computer Vision or Image Processing involve highly interac-

```
/* Example Code generated on Tue Oct 22 10:11:36 BST 1996. */

#include <Xm/Form.h>
#include <Xm/PushB.h>
#include <Xm/Label.h>

/* Declare Callback functions */
void    buttonb();

main(int argc, char **argv)
{
  XtAppContext app;
  Widget  top_wid, form,
          label0,
          button0,
          button1,
          button2;
  XmString label_str;

  top_wid = XtVaAppInitialize(&app,"X-Windows Example", NULL, 0, &argc, argv, NULL, NULL);

  form = XtVaCreateWidget("form", xmFormWidgetClass, top_wid, XmNfractionBase, 10, NULL);

  label_str = XmStringCreateSimple("Label");
  label0 = XtVaCreateManagedWidget("Label",
    xmLabelWidgetClass, form,
    XmNlabelString, label_str,
    XmNtopAttachment,  XmATTACH_POSITION,
    XmNtopPosition,  4,
    XmNbottomAttachment,  XmATTACH_POSITION,
    XmNbottomPosition,  6,
    XmNleftAttachment,  XmATTACH_POSITION,
    XmNleftPosition,  1,
    XmNrightAttachment,  XmATTACH_POSITION,
    XmNrightPosition,  8,
    NULL);

  button0 = XtVaCreateManagedWidget("Button",
    xmPushButtonWidgetClass, form,
    XmNtopAttachment,  XmATTACH_POSITION,
    XmNtopPosition,  2,
    XmNbottomAttachment,  XmATTACH_POSITION,
```

Figure 15: X Window Source Code

tive choice of parameters and values, especially when experimenting with new images to process. Experimenting with values and receiving immediate visual feedback is clearly a very useful learning tool in this area. Our Computer Vision courseware make extensive use of Java applets to facilitate such interaction. Examples using thresholding (a simple image processing task where an image is *binarised* – divided into area that are either black or white – based upon pixel values within the image being above or below a given *threshold* value) are illustrated in Figs. 16 and 17.

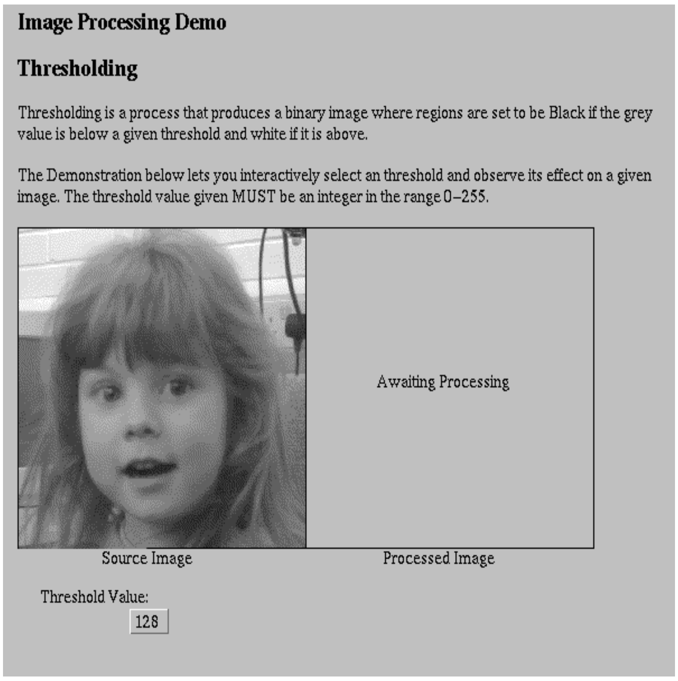

Figure 16: Initial Image Display

9 Response to Courseware

The adoption of the WWW as a courseware provider has proved to be successful. It provides a flexible way to provide a variety of presentation methods. Since programs and/or scripts can be run from a server there is little that cannot be provided on-line.

We have been able to demonstrate that this approach is possible and further more does not pose any trouble to implement given a good knowledge of UNIX programming.

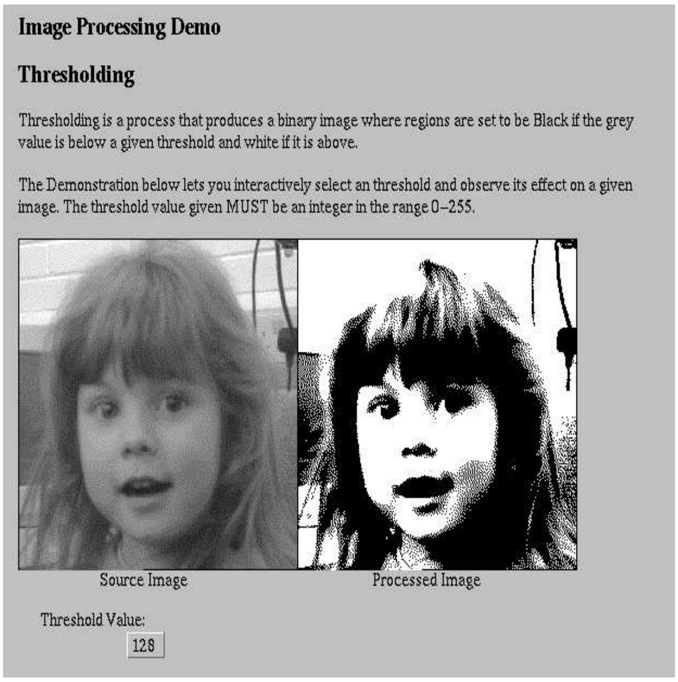

Figure 17: Thresholded Image

Previous problems of running programs and applications across the WWW appear to have been resolved with the advent of Java. This has had significant advantages in terms of implementation and speed up due to being able to handle data internally in an *applet*.

Student response to the courseware has also been positive. The courseware has been available for the whole of the vision systems course. Student response to this was good. This was evident from the use it received during the course and from student responses — both informal verbal comments and from end of course questionnaires. Very little difficulty was experienced in using the system. It also has proven popular as a revision aid during examination times.

The layout in the form of a hypertext book appears to be to favoured format for students to browse through topics and the provision of advanced key word searches aids revision and general course work preparation. The provision of source code and algorithms for image processing routines, runnable versions delivering images as output and having both available for simultaneous study was warmly welcomed.

Formal student projects and studies have also been set up with the aims of gauging student response and also to evaluate our work with relation to other WWW courseware. The results of these surveys have been presented in [12, 13]

10 Conclusions

Using a structured framework for the design of online courseware produces teaching materials which are educationally sound and easily extendable. In addition to this the materials are easy to develop and are popular with users.

11 Using Our Courseware

Our courseware is freely accessible over the WWW (URL: *http://www.cm.cf.ac.uk/-Teaching/*). We welcome comments and input on the courseware developed so far.

References

[1] F. Culwin and A.D Marshall. The design and provision of software engineering education over the web. In *Proceedings of the 5th International WWW Conference '96 (Posters)*, pages 217–226, Paris, France, May 1996.

[2] R. Gagné, L. Briggs, and W. Walter. *Principles of Instructional Design (4th edition)*. Harcourt, Brace, Jovanovich, New York, USA, 1992.

[3] W. Hall and H. Davis. Hypermedia link services and their application to multimedia information management. *Journal of Information and Software Technology (Special Edition on Multimedia)*, 1994.

[4] R. Heinich, M. Molinda, and J.D. Russell. *Instructional Media*. Macmillan Publishing, New York, USA, 1993.

[5] K. Hughes. Entering the world wide web: A guide to cyberspace. *World Wide Web Document URL: http://www.hcc.hawaii.edu/guide/www.guide.html*, 1994.

[6] S. Hurley, A.D. Marshall, S.N. McIntosh-Smith, and N.M. Stevens. Courseware for parallel computing using mosaic and the world wide web. In *Proceedings of the 2nd International WWW Conference '94*, volume 1, pages 499–508, Chicago, USA, October 1994.

[7] S. Hurley and N.M. Stephens. Courseware in high performance computing. In *Proceedings International Conference on Parallel Computing for Undergraduates*, Colgate, USA, June 1994.

[8] R. Kozma. Learning with media. *Review of Educational Research*, 61(2):179–211, 1991.

[9] D. Laurillard. *Rethinking University Teaching*. Routledge, London, U.K., 1993.

[10] A.D. Marshall. Hypertext based computer vision teaching packages. In *Proc. SPIE Conference on Machine Vision Applications, Architectures, and Systems Integration III, Photonics East 94*, volume 1, pages 207–219, Boston, USA, October 1994.

[11] A.D. Marshall. Developing hypertext courseware on the world wide web. In *Proceedings of ED-Media 95: World Conference on Educational Multimedia and Hypermedia*, volume 1, pages 418–423, Graz, Austria, June 1995.

[12] A.D. Marshall and S. Hurley. Assessing multimedia-based courseware. In *Proceedings of ED-MEDIA 96 - World Conference on Educational Multimedia and Hypermedia*, Boston, USA, June 1996.

[13] A.D. Marshall and S. Hurley. The design, development and evaluation of hypermedia courseware for the world wide web. *Journal of Multimedia and its Applications*, 3(1), June 1996.

[14] A.D. Marshall and S. Hurley. Hypertext-based courseware delivery methods for the world wide web. In *Proceedings of ED-MEDIA 96 - World Conference on Educational Multimedia and Hypermedia*, Boston, USA, June 1996.

[15] A.D. Marshall and S. Hurley. Interactive hypermedia courseware for the world wide web. In *Proceedings of ACM SIGCSE/SIGCUE Conference*, Barcelona, Spain, 2-5th June 1996.

[16] A.D. Marshall, S. Hurley, S.N. McIntosh-Smith, R.R. Martin, and N.M. Stevens. Novel uses of computers for teaching. *AXIS: The UCISA Journal of Academic Computing andInformation Systems*, 1(3):30–41, 1994.

[17] W. Montague and F. Knirk. What works in adult instruction: the management, design and delivery of instruction. *International Journal of Educational Research*, 19:327–433, 1993.

[18] D. Rowntree. *Educational Technology in Curriculum Development.* Paul Chapman, London, U.K., 1982.

[19] A.M. Zin and E. Foxley. Automatic program quality assessment system. In *Proceedings of the IFIP Conference on Software Quality*, S P University, Vidyanagarm India, March 1991.