International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:3, No:6, 2009

A Generic e-Tutor for Graphical Problems

B.W. Field

***Abstract*—**For a variety of safety and economic reasons, engineering undergraduates in Australia have experienced diminishing access to the real hardware that is typically the embodiment of their theoretical studies. This trend will delay the development of practical competence, decrease the ability to model and design, and suppress motivation. The author has attempted to address this concern by creating a software tool that contains both photographic images of real machinery, and sets of graphical modeling 'tools'. Academics from a range of disciplines can use the software to set tutorial tasks, and incorporate feedback comments for a range of student responses. An evaluation of the software demonstrated that students who had solved modeling problems with the aid of the electronic tutor performed significantly better in formal examinations with similar problems. The 2-D graphical diagnostic routines in the Tutor have the potential to be used in a wider range of problem-solving tasks.

***Keywords*—**CAL, graphics, modeling, structural distillation, tutoring.

I. Introduction

Among the multitude of difficulties facing 21st century engineering education are two that are growing more critical as time progresses: (a) pressures of reduced government funding have increased student/staff ratios and reduced opportunities for personalized tutoring, and (b) those reductions in funding, along with increased concerns about liability and safety have limited the opportunities for hands-on or otherwise realistic experiences for undergraduate engineers. When coupled with the shift toward student-centered learning (and the desire to match learning opportunities to individual needs) we find that those difficulties lead to students' perceptions of a widening gulf between engineering practice and engineering education, with scarce opportunities for connecting individuals with their future profession.

Those funding reductions are being addressed (to varying degrees of success) by a need for universities to earn income from other activities – mainly through research grants. Research success is consequently a desirable attribute for modern academics, while ongoing liaisons within the profession are less well regarded. Time, modest practical experience and funding limitations also create a gulf between many engineers in academia and industry, and this has an impact on the programs of teaching offered in engineering courses.

The author and his colleagues at the University of Melbourne have observed that the majority of engineering sciences taught at their universities are bereft of the artifacts associated with the science, and in some instances, are bereft of realistic representations (e.g., photographs, videoclips) of those artifacts. Yet we have observed that students are highly motivated by the existence of realistic ("practical") examples of the theory: it appears that an appreciation of abstractions requires some time to mature, perhaps even well after graduation for some. The experienced teacher-researcher already possesses this ability to abstract from reality, and often teaches from the abstraction, rather from the reality. This approach can lead to student dissatisfaction and their evaluation comments that a study unit is "too theoretical".

The act of constructing an abstraction from reality is called modeling. Models may be mathematical/ algebraic, physical, graphical or symbolic, or some combinations of these. When the models contain elements that are the basic building blocks of the engineering discipline, we call the modeling process that of 'structural distillation' [1]. The usual step following such a structural distillation is to find and include the numerical data that is specific to the discipline, thereby allowing the engineer to make detailed predictions from the model.

It is evident that a practicing engineer should be able to form proper and correct models, uncover the data for their particular problem, and then 'solve' the problem to meet the final need [2]. Yet there is little evidence that undergraduate engineers are schooled in the art of structural distillation. The gap in this ability became apparent to the author and his colleagues, whose specialist teaching area is in mechanical design. In open-ended design problems the student may progress in either of two directions: (a) start at a conceptual idea, and systematically prove that the concept will work, or (b) start at a calculated (numerical/graphical) descriptor of the requirements and systematically develop the physical embodiment that would achieve the requirements. In both instances the student needs to formulate a model that bridges the two elements, and this is often the most inadequately performed task in an undergraduate design [3].

Most of the conceptual modeling tasks (structural distillations) needed in undergraduate problems are very basic, and require only a few minutes of effort (from a capable student). It is impractical to personally tutor large groups of students in this undertaking. With this restriction in mind, coupled with the concern for the reduction in 'practicality' in

B.W. Field is a Professor in the Department of Mechanical and Aerospace Engineering at Monash University, Australia. (+61 3 9905 3518; e-mail bruce.field@eng.monash.edu.au).

many undergraduate engineering courses, the team conceived the potential for a computer-based tutor that could: (a) contain realistic representations of engineering artifacts (perhaps animated), (b) offer a method of allowing students to formulate structural distillations of those artifacts, and (c) implement a technique for correcting errors on an individualized basis.

Support for the development of such software was forthcoming in 2001 from a joint funding scheme created by Melbourne and Monash Universities. The basic version of the electronic tutor was written during that year, and is called MOMUS Tutor (Monash-Melbourne Universities' Structural Tutor). Somewhat perversely, 'Momus' is also a word for a faultfinder, or persistent critic, derived from the Greek god of ridicule.)

This paper describes the underlying educational philosophies and program structure of the MOMUS Tutor, and reports the educational outcomes from its first year of use.

II. Principles of MOMUS Tutor

It was intended that the proposed electronic tutor should be available for structural distillations in any of the engineering sciences in a mechanical engineering course, and it was envisaged that, since the fundamental issues being addressed by the Tutor were likely to be common across several disciplines (and outside of engineering), it might be possible to construct a fairly general tool. Since the engineering sciences tend to work in relative isolation, and students see this separation in their timetables and assessment activities, it was desirable to compartmentalize the modeling activities for each science. It was also recognized that the issues of modeling within one sub-discipline need to be tested over several different learning tasks to ensure that the principles have been properly understood.

These aspects led us to hypothesize that it should be possible to construct a tutoring program with generalized capabilities that could be customized for separate pieces of common machinery, the relevant sciences, and even the disciplines of an educational program. The common element within the Tutor software would be the capability of diagnosing the appropriateness of a 2-D image created or manipulated by the learner.

The requirements suggested a 'grid' of practice problems, with grid axes separately representing common examples of hardware and engineering sciences, with one or more tasks made available in each of the grid elements. Fig. 1 is the first grid for MOMUS Tutor that was coded. We intended that problems would be generated in each of the grid elements, and students might decide to select individual problems, sets of problems from an engineering science (a column), or the issues associated with an artifact (a row). The team was especially interested in the formulation of problems along a row, since this should illustrate the integration of several engineering sciences within the design of a single machine (one of the primary purposes of an engineering design unit).

The machines themselves were intended to be realistic representations, so were likely to be information-rich (constructed from photographs, videotape, or rendered 3-D models), causing the artifact representations to occupy large volumes of electronic memory. To minimize the difficulty of managing such large amounts of memory, it was also desirable that the machines be suitable for modeling tasks in several sciences. At present, it is expected that final versions of the MOMUS Tutor for mechanical engineering, with eight different machines, will fit onto a conventional CD ROM, or be available through campus networks. It may not be suitable for dial-in modem access.

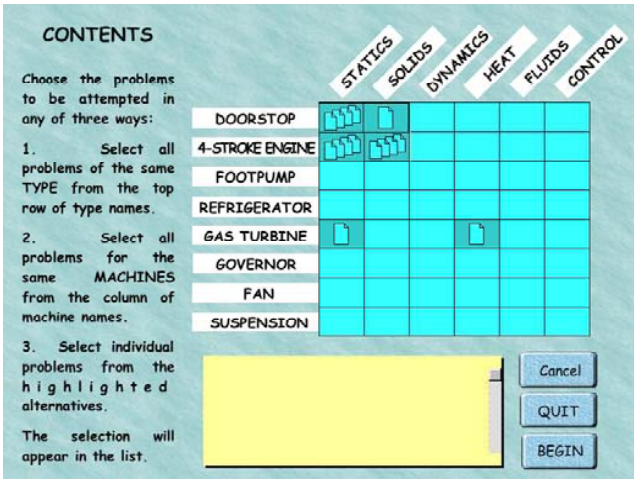

Fig. 1 Contents page of MOMUS Tutor with several problems available

MOMUS Tutor was coded in Macromedia's® Director, a common base for educational software. It facilitates the simple creation of a stand-alone 'projector' for distribution, and compressed versions to be played by Shockwave (free downloadable software from Macromedia®. A starting 'movie' allows students to jump to an introduction, describing the purpose of the software and how to use it, or go directly to the problem set. From the contents frame (Fig. 1) one or more problems can be selected, then attempted in sequence. The hardware is represented in separate 'frames' of the movie, and the separate sciences are represented by independent 'casts' [4] of icons.

III. Coding MOMUS Tutor

The core program structure is shown in Fig. 2, with a succession of 'frames' running indefinitely from left to right. Normally, a 'movie' begins on the left and progresses automatically to the right, except where coding causes the movie to freeze.

When the user enters the frozen 'Problems' screen (Figs. 1 and 2), MOMUS Tutor explores the folder in which it is located and identifies any compatible text files (those with names that include a hardware title, a science cast, and a number from 1 to 9). The Tutor then constructs the 'contents

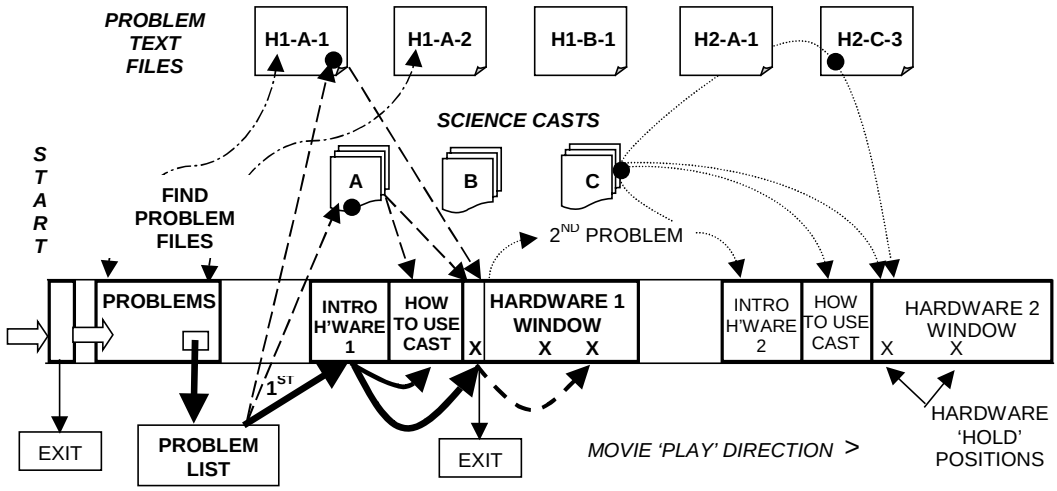

Fig. 2 Structure of core MOMUS software

grid', and places a single problem icon in a grid location where one problem has been found (such as with 'Gas Turbine-Heat', in Fig. 1), or a multiple-problem icon where two or more problems have been found (such as with 'Doorstop-Statics' in Fig. 1). Numbered pop-up lists for the multiple-problem sets are constructed. The user can select one or more problems from the grid, then 'BEGINs' the first task.

MOMUS Tutor then opens the appropriate *problem text file* and loads the task text and problem settings into the hardware window. The chosen *science cast* is loaded into the hardware window, and a cast instruction page is loaded into a layer above the hardware window. Control passes to a hardware introduction window.

The introduction to the hardware item may be several 'frames' long, and typically contains information about the hardware that is to follow. In the case of the '4-stroke engine', the information frames contain a simple animation of the 4-stroke engine cycle.

The student can progress to the 'how to use cast' frozen window to learn how to manipulate the cast icons available in the problem, or go directly to the first problem's window, and then to any frozen position in the animated hardware movie.

Thereby, a problem is constructed from the combination of an artifact with the contents of a science cast, plus the specific text that is authored for each problem.

The potential to animate machines (in order to gain insight into their functionality) and the need to isolate portions of a machine in some modeling processes led to the decision to 'construct' the machine representations from separate images, 'assembled' together to show the whole device. Fig. 3 is the screen image of the first problem in the grid shown in Fig. 1, and the hardware is 'made' from three photographic images: the bent doorstop leg, the doorstop bracket, and the door/floor. There is also a background image to give context.

Students are able to construct the line-diagram models that represent the machine or selected portion of the machine by dragging and dropping segments of the model onto the appropriate part of the image(s). A typical tutorial problem would ask a student to construct a line diagram model for some part of the image under defined external conditions (e.g., loads, temperatures, speeds) that might be used in the solution within a particular engineering science (e.g., dynamics, thermodynamics, control). Their answer will comprise several components, including the machine configuration at some point in its cycle, the highlighted components, and the locations, shapes and alignments of the various modeling 'icons' that define the model. The answer is therefore essentially a unique 2-D image, the construction of which is rendered in a convenient manner by the Tutor's interface.

When the student has completed the task, the 'Next' button identifies the second problem on the selected list, and goes through the same loading routine for that problem.

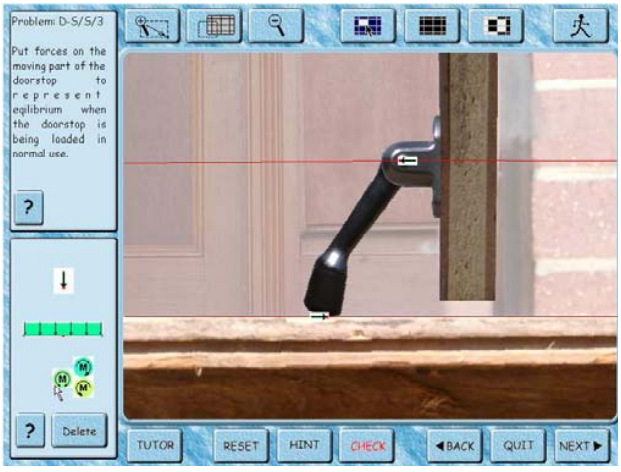

Fig. 3 Appearance of MOMUS Tutor hardware screen during an attempt to solve a problem

Fig. 3 shows the screen of the Tutor during the formulation of an answer to a basic problem in statics. The object (in this case a simple doorstop) fills the main window. The task is

defined in the upper left-hand window, and the modeling icons (point and distributed forces, and moments) are available in the lower left-hand window. When the Tutor offers feedback after a student asks it to 'Check' the answer, a feedback window overlays modeling icons. The top row of buttons allows the image to be manipulated – zoomed, selected and animated, and the lower row of buttons allows the problems to be navigated. The student's answer (in the case shown in Fig. 3) is two copies of the point force icon dragged, dropped and rotated onto the image, which may contain several de-selected parts of the machine, defining the 'free-body boundary'. The alignment of the force icons is facilitated by the creation of internally generated straight line 'icons' that can be gripped and rotated.

The Tutor is programmed to diagnose the student's answer, and then to offer appropriate comments that have been prepared in advance by the educator who set the problem after switching the software to an authoring mode.

IV. Diagnosing Student Answers

MOMUS Tutor contains a generic diagnostic routine that compares the screen appearance with previously stored solutions.

While authoring, the educator has the opportunity to create a number of possible solutions – correct or incorrect, that are judged to be likely responses by students. The first solution that the educator creates is defined as the 'target' solution (the most desired correct solution), but any successive solutions loaded into the Tutor can be examples of the most common types of errors that students tend to make. For example, the solution shown in Fig. 3 was an incorrect solution that was offered by 20% of the students who attempted the problem when it was set on paper as a 'spot test'. This approach follows a similar philosophy to that adopted by Scott and Stone [5] with their introductory Dynamics tutor at the University of Western Australia and their generalized 'Jellyfish' tutorial environment.

The science icons in MOMUS Tutor have definable characteristics that can be separately enabled when the science cast is constructed. For example, the 'Moment' icon with the 'M' in Fig. 3 is called a *free vector*, and has the same physical effect on an object wherever it is applied onto that object. Therefore the location of the 'M' is characterized in MOMUS by the code number of the machine component over which the icon has been placed. The 'point force' (two of which have been placed in Fig. 3) is characterized by the location of the point (the tip of the arrow) over the machine and its slope (the two forces in Fig. 3 are horizontal, but facing on opposite directions), so is characterized by the screen position in x and y directions, and its slope, each with a definable ± tolerance. That is, three parameters are diagnosable in a point force. The distributed force (the central selectable icon in Fig. 3) is diagnosed as for the point force, plus its (stretchable) length: a total of four diagnosable characteristics.

Using the set of authored 'solutions', the diagnosis in the MOMUS Tutor is performed in two stages.

First stage: The diagnosis conducted by the Tutor is a search through its set of stored solutions for a close match (within tolerances), and, if it finds a match, the Tutor offers the corresponding feedback comment that was pre-stored along with that solution.

Second stage: If a close match to the student's answer is not found, the Tutor uses its second diagnosis routine. In this routine the Tutor compares successive elements of the student's answer with the 'target solution', and offers feedback associated with the first substantial mismatch that it finds. These feedback comments are also pre-stored when the educator set the problem, and cover, in order, circumstances where:

1) No icons have been placed,
2) The machine is wrongly configured (wrong 'frame'),
3) The wrong machine components have been selected,
4) At least one wrong icon has been placed,
5) At least one correct icon has been wrongly placed,
6) At least one adjustable icon has been wrongly sized,
7) At least one rotatable icon has been wrongly aligned,
8) At least one icon is missing.

In the case of circumstances 4 to 7, an offending icon can be authored to flash.

The Tutor keeps track of the number of times that an identical 'error' occurs, and provides access to second and third level 'hints' that the educator has prepared. The student has no direct access to the 'target' solution, or any other 'good' solutions that have been stored, so the hints and feedback have to be constructed by the educator to direct students toward the target, and the target solution needs to have a feedback comment that identifies itself as the termination of the problem. In this way it was intended that the Tutor could follow a similar structured approach to that of an experienced human personal tutor.

V. Authoring in MOMUS Tutor

The access point for problems in the Tutor is a 'contents page'. This page (Fig. 1) displays a grid, where the rows represent the alternative 'machines' available for analysis. The 'doorstop' in Fig. 3 is the first of these machines. The columns represent the engineering sciences for which problems may be authored. The 'static equilibrium' icons in Fig. 3 belong to the first of the engineering sciences. It is therefore possible to set or access problems in any of the nominated engineering sciences applied to any of the machines, by selecting the corresponding grid element. The Tutor can be used to create, then access up to nine problems in each grid element, although it begins with a completely empty grid. Currently the grid is 8 machines x 6 sciences, allowing a teacher to create up to 8x6x9 = 432 separate problems.

After entering the authoring mode, protected by a password, the educator can select any of the grid elements to create or edit a problem. This route is shown in Fig. 4. The starting configuration is then chosen: image size, scales, default

sensitivities (tolerances) for the diagnosis, the problem text and the subset of modeling icons, including any pre-placed icons if desired. The feedback comments associated with the second stage diagnosis are then entered, followed by the target configuration and its specific comments. The required diagnostic accuracy (tolerance) for this and other sample solutions can be set for each solution by manipulating visual 'tolerance zones'. For icons that can be rotated, the tolerance zones for the alignment are shown as sectors of a circle, such as those shown as dark 'wedges' associated with a 'beam' icon in Fig. 5: the Tutor will accept any alignment of the icon that falls within the sector. The tolerance zones for positions of icons in x-y space are rectangular areas, such as those associated with the three forces in Fig. 5. Other icons may have special characteristics: the length of the beam in Fig. 5 can be no less than that shown, but could be larger (with redundant overhang), so its tolerance zone for length is indefinitely long each side of a central minimum length.

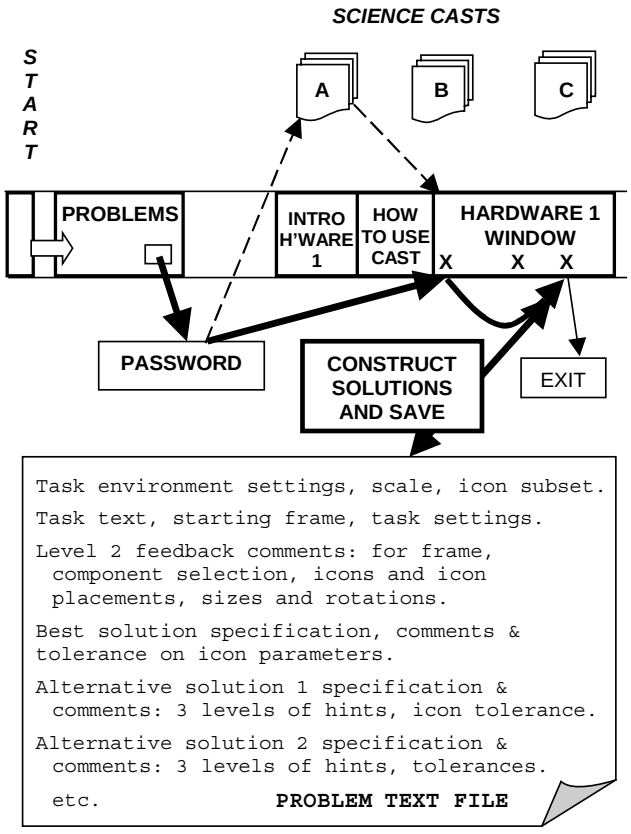

Fig. 4 Authoring a problem: after a problem grid is selected, the cast is loaded into the hardware movie. Input windows then allow the author to configure the hardware, then create solutions and write appropriate feedback comments. The outcomes are written to a text file and stored in the MOMUS folder.

Any number of alternative solutions and their feedback comments are then entered. The problem-setting task is then terminated, and all of the information about the problem and its solutions is recorded in a separate text file, averaging 35 kilobytes in size (and easily transmitted through the internet, even to students with slow a dial-up internet connection).

Subsequently, when the Tutor is opened, it searches its default directory for problem text files, and, finding any, makes them accessible in the contents page. In the authoring mode, any existing problem can be edited or extended: in the tutoring mode, each problem can be selected individually, or in sets, and attempted by students.

Fig. 5 Authoring a problem, showing the colored tolerance zones associated with machine element icons (beam and point forces in this case)

VI. Evaluation of MOMUS Tutor

The doorstop images of Fig. 3 were created, along with the set of static equilibrium icons. Coding for most of the desired characteristics of the software was completed, including the methods for manipulating the images, manipulating the icons through pop-up selections, rotations, and distortions, diagnosing the answers, and authoring new problems. It was therefore possible to prepare up to nine problems in statics with one piece of machinery, and to test the software with students. The Tutor was mounted onto fileservers at Monash University, and onto a smaller number of stand-alone PC's at the University of Melbourne.

The evaluation comprised the following sequence:

1) Conceive a problem that could be set in the MOMUS Tutor, and set the problem on paper.
2) Administer the problem to students at the respective universities, allocating credit points for correct solutions. These problems, which only required the placement of two or three force images, were constrained to allow only five minutes of effort.
3) Collect the alternative solutions and group them into identical (or near-identical) sets.
4) Code the most common of the sets of solutions into the Tutor, along with the associated feedback comments. (Across the four initial problems generated, there was an average of 18 different sets of solutions coded per problem. This coding took an average of 1.5 hours per problem, following an average of 1.5 hours to define each set from the 300+ students' attempts on each set).

5) Encourage some students to seek the solutions to the problems via the Tutor.
6) Administer a similar problem, and dissimilar problems involving the same principles, to all students, and seek differences in the success rate between students who had used the software, and those who had not.

A. Results of the evaluation

Four different problems on the equilibrium of the parts of the doorstop machine were administered on paper during the trial. These included the basic, 2-force single moving part through to the more complex three-force 2-part doorstop assembly. Approximately 300 students at the two universities attempted these problems simultaneously. Because of earlier experiences [6] the team was not surprised at the low success rate of their students: only 1%, that is, five of the students created correct solutions to all four of the tasks.

The most common solutions for each problem were coded into the Tutor and students at Monash University were given access to those solutions one week after they had attempted the problem. For a variety of reasons, only a few students took the opportunity to explore the solutions and find out how well they had performed, or to seek the 'correct' solution.

Following the fourth problem, a fifth test problem in equilibrium was set, representing an abstract 2-piece object with one external load, and two support points. The abstract object could be analyzed with exactly the same force images as were used on the four tasks set on the doorstop, but the fundamental similarity was not immediately obvious to most students. Students were also asked to indicate how much time, if any, they had spent using the Tutor software during the previous month.

Although only ten of the 120 students at Monash University indicated that they had used the Tutor, five (50%) of this group reached the correct solution for the fifth problem, whereas only 5% of the remainder of the group did so (consistent with their capabilities found in earlier tests). This was not conclusive evidence that the Tutor had increased student skills in the area, but at least the results were encouraging. An alternative explanation: the self-selection of students who used the Tutor may have biased this group to contain more educationally motivated students, who may well have found alternative sources of learning. Ethical and administrative obstacles precluded the authors from using fully randomized groups.

In the main evaluation study, students at the University of Melbourne were not given access to the Tutor until the classroom tests had all been completed. However, their final examination in the design subject was to include a fifth doorstop equilibrium problem, another more abstract problem in static equilibrium (comprising a multi-segmented loaded ring), and a set of questions relating to their use of the Tutor. Four more potential doorstop problems were coded into the Tutor, making a total of eight, and students were told that one of the four new problems would appear on the examination. None of those four new problems contained the correct solution or useful comments if students attempted to solve them in the Tutor. We expected that some students would try to use the Tutor on the first four problems before they accessed the four new problems, but we thought that some students would rely on others to 'find' the new problems for them, and therefore not access the Tutor at all.

The examination results were analyzed to distinguish the achievements of those who had used the Tutor from those who had not. The results indicated a significant correlation of 0.33 ($n=182$, $p<0.001$) between the number of problems solved using MOMUS Tutor and success on the examination problem. Cross correlations with other possible causes for differential performances were not as significant. (For example, there was weak correlation between the success on the examination problem and success on the test problems throughout the semester, and between success on those test problems and usage of the MOMUS Tutor). Other relevant correlation coefficients are summarized in Table 1. It was concluded that the most likely cause of better examination performance was the successful exposure to problem-solving with the Tutor software.

TABLE 1
CORRELATION COEFFICIENTS FOR MOMUS TUTOR EVALUATION

	Mean	S.D.				
Doorstop examination problem result / 8	2.5	1.2	**0.33**			
Number of problems attempted on MOMUS / 4	2.8	1.2	0.20	**0.5**		
Time spent using MOMUS Tutor, minutes	40.6	47.6	0.14	0.00	0.20	
Score on similar doorstop tests in class / 2	0.4	0.4	0.00	0.15	0.00	0.20
Final mark gained in Engineering Design course / 100	60.1	11.5				

N=182. The correlations in bold are significant ($p<0.001$)

B. Discussion and comments

The encouraging findings from the evaluations of the Tutor led to minor refinements in the diagnostic routines and the expansion of the hardware and icon sets to include a four-stroke engine and the elements used for representing columns, beams, shafts and tensile members (Fig. 5). The engine image can be animated continuously, or stopped in various critical configurations. By de-selecting (ghosting) external components, such as the crankcase, images of all the important separate parts can be seen, selected and magnified for detailed study. These new aspects to the Tutor allowed us to generate both static equilibrium and structural elements for two pieces of hardware (i.e., four grid elements in MOMUS's contents).

VII. Unexpected Expansion

In separate projects, the author and senior undergraduates have identified hardware that would be motivating to lower level students, and have identified the types of imaging tasks that students found most difficult. These tasks have included the subtleties of dynamic and kinematic analysis, the selection of manufacturing processes, the construction of numerically-based (i.e., scale) diagrams and tutoring in descriptive geometry: the latter two of which were not considerations when MOMUS Tutor was conceived.

A. Dynamic analysis

Two honors-level project-students at the University of Melbourne surveyed their colleagues on their study difficulties, and identified a desire to more fully understand the way in which analytical analyses of dynamic (i.e., machine motion) problems should be set up so as to simplify the subsequent numerical work. The students then prepared a 'Dynamics' cast and a series of problems using the 4-stroke engine hardware (Fig. 5) that the author had created earlier.

Since the procedures for setting up the analytical framework involved a large number of ordered concepts, such as the identification of relevant moving parts, their types of motion, a suitable analytical reference frame and so on, the students decided to utilize the 9-question option for the relevant contents grid element to create a set of sequential tasks. For every problem after the first, the 'solution' to the previous problem was pre-placed on the model, and the next task was set. Fig. 6 is the problem screen for the fourth problem in the sequence.

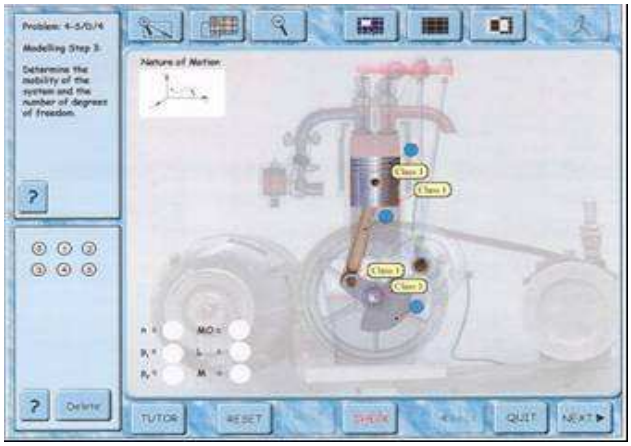

Fig. 6 Fourth problem in a structured sequence in Dynamics, with several pre-placed dynamics icons. The student is required to select, drag and drop an icon into each of the targets in the lower left of the hardware window.

In Fig. 6, the 4-stroke engine is presented with the relevant moving parts highlighted. The labels, the upper left illustration, and the targets on the lower left have been pre-placed for this problem. The animation button (upper right) has been disabled and the engine has been frozen in its current position. This was the first group of problems set in MOMUS in which a student could 'give up' and find the correct answer (by simply going to the next question). The evaluation of this task suggested that it could be appropriate to provide some form of 'show me the right answer' option, possibly by enabling an additional button that constructed the preferred answer after a set number of failed attempts, or multiple similar failures.

B. Manufacturing analysis

Two honors-level students chose to adapt MOMUS to an unanticipated science area, namely that of determining the appropriate manufacturing methods for metal components. Neither of the existing two MOMUS hardware items was suitable, so the students obtained a motor mower engine, dismantled it, and generated a series of photographs of the parts in a new movie sequence. A succession of problems was authored to jump to, and freeze on the appropriate image. The manufacturing science icon set comprised simple select, drag and drop icons for which the author had already coded suitable diagnostic routines. This student project demonstrated that naïve programmers could easily create a set of hardware images and swap them into MOMUS without difficulty. The screen for this version is shown in Fig. 7.

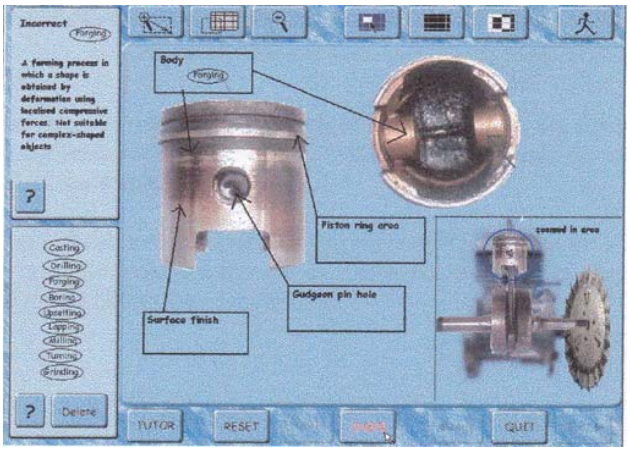

Fig.7 Student-created MOMUS hardware including selectable translucent engine parts and separate photographs of an engine part. The selectable icon set comprises nine drag-and-drop labels, and the drop areas are defined by four pre-placed targets.

C. Descriptive geometry

The potential versatility of the Tutor as a diagnostic machine for 2-D graphic problems, together with the perceived learning difficulties for students undertaking studies in descriptive geometry encouraged the development of non-modeling modules in the MOMUS Tutor. Descriptive geometry involves the solution of three-dimensional spatial problems by manipulating a series of 2-D 'projections'. The projections are typically presented as abstractions that are usually separately ambiguous, but which make sense when families of these abstractions are assembled. Many students have difficulty in learning how to 'read' these projections and

subsequently create additional views (that may be required in order to solve some real or abstract problem). In its simplest form, students are supplied with two related views of an object view (classically referred to as *plan* and *elevation*, but these terms may have little meaning in some problem tasks), and are required to construct a third. MOMUS Tutor has now been programmed with a set of nine problems that take students from a simple introduction, which includes a 3-D image to aid visualization (Fig. 8) through to visually difficult images. In all cases, the solution is constructed by placing multiple copies of a simple drag-and-drop line component. The lines may be rotated and stretched to complete the image.

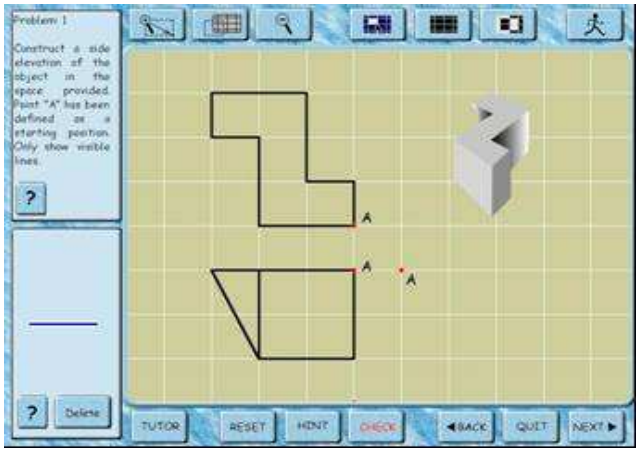

Fig. 8 MOMUS Tutor screen with an introductory problem in descriptive geometry

For this series of problems, a single background grid was prepared, containing six, vertically aligned line drawings (the first two of which are seen in Fig. 8) and one 3-D image. Two icons were made: the single straight line that was stretchable and rotatable, and the datum point 'A' to be pre-placed onto the grid to define the required projected view for the problem. In successive problems, the 'A' could be positioned in one of three locations for the same starting images, and/or a different pair of line drawings could be presented in the frozen frame

Although purpose-coded tutoring software in descriptive geometry has been written with an easier interface for drawing lines [7], the MOMUS Tutor has a special advantage. The coding of this task and the basic problems only required a few days of work, since the 'behaviors' of the icons and the versatility of being able to pre-set the hardware image and pre-placed icons had already been coded. In the current version, six alternative pairs of initial views are available, with four alternative placements of the datum 'A', meaning that 24 alternative problems are possible. Many more alternatives would be available by constructing other sets of starting images (perhaps in later 'frames' of the movie), and other types of lines could be readily added to the cast: dashed 'hidden' lines and stretchable curves could readily use existing diagnostic options in MOMUS.

D. Numerically-based problems

Both faculty and students reported difficulties in learning how to analyze the loads on structural beams, and in particular how determine the effects of those loads at various positions along beams. Classically, this type of analysis involves the production of graphs showing the variations in two loading parameters (termed SF and BM) as functions of beam position. These graphs need to have correct shapes and sizes. It would have been quite straightforward to use the graphical construction tools in MOMUS to create an icon set for drawing these graphs, but some alternative approach was needed if numerical values were to be chosen by the learner, since this characteristic was not planned to be a diagnosable property in MOMUS.

In providing a MOMUS module for learning how to analyze beams, the requirement for checking numerical values was addressed in two separate ways. The first method was to provide graphical axes and stretchable icons so that the size of a one-dimensional icon could be set to a chosen measure, and the second method was to provide an expandable icon with its computed size superimposed for feedback. In both instances, the actual size of the icon could be diagnosed with existing MOMUS codes. The screen for the first of nine 'Beams' problems is shown in Fig. 9.

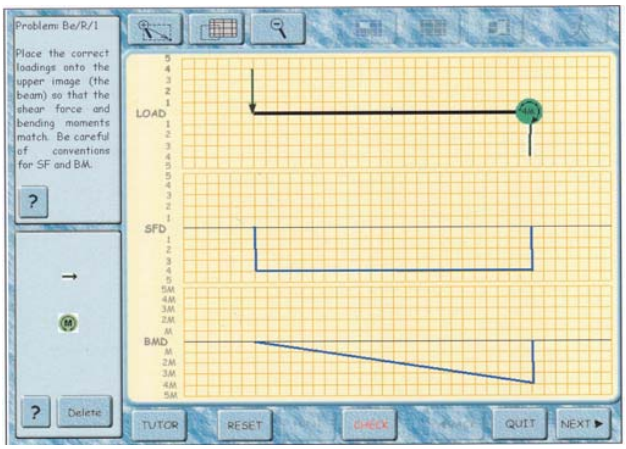

Fig. 9 MOMUS screen for the solution to the first problem in 'Beams'. The two graphs have been given, and the task requires the selection, sizing and replacement of three loading icons on the top diagram.

Two copies of the 'force' icon have been placed (rotated to face in opposite directions) and stretched to be four 'units' long (this correct length can be worked out from the *SFD* graph). One copy of a 'clockwise moment' icon has been selected from a pop-up pair (from the same 'M' icon as in Fig. 3) and placed. This moment icon has been overlaid with a second icon, in which an item of text can be inserted. The moment icon has been made expandable by the use of lateral 'handles', and the expanded size has been converted to an integer number, overwriting the text in its overlay. In this case the expanded size is correctly labeled '4M' (a fact that learners can deduce from the *BMD* graph).

E. Remote monitoring of learner behavior

There are two particular concerns when MOMUS is being prepared for a learning task: (a) what common errors will the learners make, so which incorrect solutions should the teacher code, and (b) how do we know that the feedback offered by the teacher in response to an anticipated error is not misleading?

For the problem set involving the 'Doorstop-Statics' problems (Fig. 3), each problem was printed on a sheet of A4 paper and given to more than 100 students as a 'class test'. Their attempts were classified, and the most common answers were coded into MOMUS. For example, the problem shown in Fig. 3 has one correct, and 13 incorrect solutions in its text file. This approach is effective, but somewhat laborious.

Normal CAL materials are evaluated during the development phase by observing and surveying a sample of users. The software is then refined before it is finalized. However, MOMUS begins as an authoring tool, and will be accessed by a teacher as a 'blank' page. The teacher creates each problem task, and the feedback is coded as a reflection of the teacher's personal style. The evaluation of each problem, and the measuring the effectiveness of the feedback comments would be a substantial task if it had to be done for every problem that was authored.

The first of the two concerns is being addressed by using MOMUS to offer on-line tests. The version on the Monash University fileserver has an additional password option for authors: namely the creation of a 'test'. The text of a test file has no solutions or feedback comments. In the MOMUS Contents screen (Fig. 1) a test problem becomes separately colored and contains the prefix 'T'. If a student selects this file, the MOMUS screen is normal, except that the 'HINT' button is disabled and the 'CHECK' button becomes a 'SUBMIT' button. This button sends the student's solution and their personal details, to a master file.

An on-line 'Agent' 8], [9] will capture student test submissions generated from within MOMUS Tutor, and feed them in a grouped form (with similar submissions grouped together) to the educator. Student attempts may then be assessed, and accessed through the 'edit' feature in MOMUS Tutor, where the feedback comments can be appended, and the outcome recorded as a conventional tutorial problem text file. When students then re-visit the test problem as a tutorial task, they will have the opportunity to work toward the solution in the established manner.

The second concern with authored problems, namely an evaluation of the usefulness of any feedback comments, is expected to be addressed by a refinement to the 'Agent' used for collecting the test submission. In this case the modified Agent would capture the screen configuration (*zoom*, *pan*, and *hint*), as well as the 'solution' and a time stamp, and transmit the data to a central file every time the 'CHECK' button is pressed. This would allow the educator to follow the logic path taken by a learner, and to identify where the path to a solution appeared to deviate from the expected route. The teacher would then be able to add intermediate solutions to the set of student responses so that more efficient learning would take place. The overall aim of this latest MOMUS Tutor development effort is to reduce the gulf that exists unavoidably between educator and student in computer-mediated learning environments.

The combination of using MOMUS to collect test submissions, grouping similar submissions with an Agent, preparing tutorial problems based on the test results, and monitoring tutorial work with a similar Agent is shown in Fig. 10.

F. Expansion beyond Mechanical Engineering

The encouraging outcome from the evaluations has led to a separate project at the University of Melbourne to use the basic shell of the MOMUS Tutor with a special set of photographic images of various mechatronic devices, along with new drag and drop labels, to provide an introduction to the separate discipline of Mechatronic Engineering. There is a long-term plan to extend this approach into other engineering and non-engineering disciplines where convenient customization, author accessibility, and immediate student feedback on modeling tasks are desirable educational goals.

This example, albeit from within an area related to the discipline of mechanical engineering, nevertheless illustrates that MOMUS Tutor is customizable to a range of graphical tutorial problems, and is therefore likely to find applications in other disciplines.

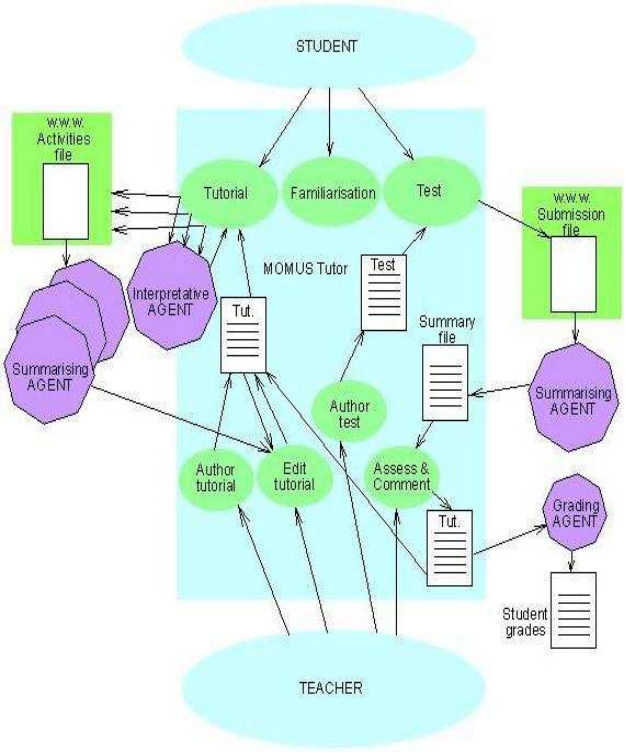

Fig. 10 Plans for the use of Agents to collect MOMUS-based test submissions (right side), convert them to tutorial problems (central), and then monitor tutorial work (left side) to identify inadequate authoring.

VIII. Conclusion

The electronic tutor gave valuable learning experiences to the students who used it in the solution of classic problems in static equilibrium, and assisted in improving a universally weak skill. The expansion of the Tutor to include a wider range of modeling icons, more exciting machinery, and abstract graphical problem-solving, is under way.

The basic diagnosable characteristic of 'select-drag-and-drop into position' has been utilized by senior engineering students, who had no prior knowledge in MOMUS's coding language, to create a wide variety of task scenarios with very little effort. This indicates that the basic shell of MOMUS Tutor might be economic for other disciplines where 2-D representations of tasks and solutions are appropriate, and where resources available for coding stand-alone CAL material are restricted.

The special characteristic of MOMUS Tutor, where a teacher can construct a series of problems from a pre-set environment, and then author any number of correct and incorrect answers along with the most suitable feedback makes MOMUS a versatile tool for on-campus and remote learning applications.

Acknowledgment

The author acknowledges the support and input from Colin Burvill and John Weir, lecturers in the Department of Mechanical and Manufacturing Engineering at the University of Melbourne for their assistance in evaluating the electronic tutor, for their suggestions on the manuscript, and for their co-operation in the development of the software. Several undergraduate students at the University of Melbourne contributed to the creation of unique modules within the software; these were Leonard Kar Yui Tao, Mazen Afara, Nazid Ab Razak, Billy Tabourlos, and Janet Karroum.

References

[1] A.E. Samuel and J.G. Weir, *Introduction to Engineering Design.* Oxford: Butterworth Hinemann, 1999.
[2] A.E. Samuel and J.G. Weir, "The acquisition of wisdom in engineering design," *Instructional Science*, vol 20, pp. 419-442, 1991.
[3] E.S. Ferguson, *Engineering and the mind's eye.* Cambridge, MA, MIT Press, 1992.
[4] G. Rosenweig, *Using Director 8*. USA, QUE, 2000.
[5] N. Scott and B. Stone, "A flexible web-based tutorial system for engineering, maths and science subjects", *Global Journal of Engineering Education*, vol 2, no 1, pp 7-16, 1998.
[6] B.W. Field, C.R. Burvill and J.G. Weir, "Student misconceptions in engineering design", *Proceedings of the International Conference on Engineering Design '01 (ICED'01)*, Glasgow, 2001, vol 1, pp 253-260.
[7] L. Murch and B. Woolfe, *The engineering drawing tutor*, University of Massachusetts, http://althea.cs.umass.edu/ckc/34spacialreason.html.
[8] T. Juan, A. Pearce, and L. Sterling, "ROADMAP: Extending the Gaia methodology for complex open systems", *Proc First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS2002)*, Bologna, 2002, pp.3-10.
[9] A. Baylor, "Agent-based learning environments for investigating teaching and learning", *Journal of Educational Computing Research*, vol. 26, no.3, pp. 249-270, 2002.