International Journal of Engineering Education Vol. 28, No. 5, pp. 1127–1140, 2012
Printed in Great Britain

0949-149X/91 $3.00+0.00
© 2012 TEMPUS Publications.

Software Environment for Learning and Knowledge Assessment Based on Graphical Gadgets*

ZELJKA MIHAJLOVIC and MARKO CUPIC
University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Electronics, Microelectronics, Computer and Intelligent Systems, Unska 3, 10 000 Zagreb, Croatia. E-mail: zeljka.mihajlovic@fer.hr

In this paper we present a successful implementation of a new software environment for learning and knowledge assessment. We introduce a new kind of gadget that is appropriate for inclusion in learning environments. The proposed gadgets are based on individualized interactive graphics tasks that are similar to educational Java applets. However, the usual graphical applets are not individualized: they do not have a defined goal for the learner and they do not evaluate the solution to a given graphical task. Using several examples, we present the development of the gadgets. The development of gadgets must address two problems: how to create and present an individualized task to the student and how to evaluate the student's solution, especially when multiple solutions are possible or solutions can be partially correct. The tasks that are described in this paper are intended for learning and knowledge assessment in computer graphics courses. However, the proposed concept is applicable to many other courses and educational activities. To study the influence of the proposed software environment on student learning and performance, we provide experimental results from several years of student use of different gadgets in preparation for mid-term and final exams and we analyze and discuss the findings. In addition, each year a selected group of students was given the opportunity to develop and implement several gadgets that were then used by their peers for learning and knowledge assessment. We discuss the results obtained and the experience gained.

Keywords: educational technology; computer graphics software; unsupervised learning; programming environments

1. Introduction

The knowledge assessment of students in undergraduate and graduate courses in many disciplines, especially engineering and computer science, plays an important role in the learning process. In general, the teacher's goal is to enrich the students' knowledge, improve the students' ability to reason and stimulate the students' creativity. The traditional approach to knowledge assessment includes paper-and-pencil methods, where the possibilities of presenting the tasks to the students are limited by a fixed representation on the paper, and no interactivity is possible.

To overcome such limitations, we propose a new approach to knowledge assessment that includes interactive and individualized tasks presented and evaluated by a computer program. Such tasks will be denoted 'gadgets'. The proposed approach based on gadgets offers interaction with graphically presented tasks and facilitates exploration of the presented problem. It enables immediate feedback offering the correct answer, provides objective evaluation, and follows new technology trends.

Currently available commercial or open source [9] solutions for course management and knowledge assessment, such as BlackBoard [2], Moodle [30] and others [17, 19], provide a usable, but limited set of features, mostly in the form of multiple-choice questions. There are also solutions such as QuizPACK [4, 5], which supports individualized tasks and which can generate and evaluate parameterized questions in the domain of the C-programming language. Another example is in de Sande [34] for the signal processing domain. An approach to adaptive testing of cognitive skills using parameterized questions is described in [6].

A further step towards the sophistication of learning and knowledge assessment is personalization [10], which enables users to find relevant learning resources in distributed knowledge repositories. Personalization requires the creation of a student profile and standardization of learning information resources [11], interoperability and portability. Notable examples of such standards are: Learning Object Metadata (LOM) [26] developed by the LTSC [28] and similar specifications from IMS [21] and PAPI Learner [31]. One requirement that we set for the software environment for learning and knowledge assessment is that it should support individualization, but not necessarily personalization. In order to support personalization, the proposed technology should have access to the student profiles, formal course description, and the relationships between gadgets and course topics. However, the proposed technology can be relevant for designing personalized software environments in the future.

To ensure a practical experience during the educational process, access to more sophisticated course laboratories, project-based learning [29] or product-based learning [24] should be provided.

* Accepted: 22 April 2012.

Practical experience is very important for future engineering professionals [35]. Therefore, to reduce the cost of expensive laboratories, many remote laboratories, in which real-world devices are connected to the Internet, have been developed [22, 25] and collaboration using remote laboratories has been considered [25]. Simulators of particular pieces of laboratory equipment are also commonly used [1]. More recent approaches to the development of learning environments try to implement game-based learning, which combines problem solving with a game environment [12, 16] and collaborative mobile learning [3]. Important aspects in game-based learning are: reaching certain goals, competition, and winning points. Remote laboratories and game-based learning environments offer students the chance of interactivity and exploration of lab equipment or game environment. It is our intention to incorporate similar concepts in our gadgets.

Most of the interactive learning environments are based on widely used technologies, such as VRML [37], X3D [38], Flash [12], and Java applets [7, 8, 20, 23, 27]. Since Java is widely available on a variety of platforms and is a well accepted technology, we decided to base our gadgets on Java. Compared with VRML and X3D, Java offers much more graphical programming options, and compared with Flash it has a larger developer base and it can be used for both the client-side and the server-side of an application.

New trends in the development of learning materials, technology and overall learning environments have also provoked the emergence of new concepts in knowledge assessment. To achieve sustainability, in addition to the need to constantly update course materials and keep up with the challenging frontiers in education, knowledge assessment should not be neglected. In addition, the knowledge assessment process should be made more attractive for both learners and teachers.

In this paper, we propose and describe the concept of software environments based on graphical gadgets that are suitable for both learning and knowledge assessment. The proposed gadgets combine the ideas of previously presented approaches to learning: they offer the ability to individualize presented tasks, they offer the ability to include simulators, and finally they draw ideas from game-based and problem-based learning, such as interactivity and a rich graphical environment. These gadgets provide individualized tasks and can be used either for knowledge assessment or for learning and experimentation, which is an important benefit when compared with the more traditional approach that differentiates learning materials and knowledge assessment as two separate categories. Besides exposing students to gadgets from a consumers' standpoint, we propose engaging the students in the development of the gadgets, and we verify that the engagement will have positive effect on the students' learning outcome. Additionally, by being included in gadget development, students will gain practical experience.

We have stated several hypotheses on gadget use and gadget development, such as: that learning via the use of gadgets leads to similar or better learning results than with traditional methods; that the interactive graphical gadgets are more attractive to students than the textual ones, and that the homework that involves gadgets can have a positive impact on student performance in the mid-term exam that follows the homework. The verification of the hypotheses is based on statistical analysis of the data collected and a student poll. Although the proposed gadgets can be used for a variety of purposes, in this paper we will present their successful application to teaching computer graphics. For this reason we believe that the proposed approach could be especially interesting for computer graphics educators.

In Section 2, we discuss related work and current situations involving interactive learning materials in computer graphics. In Section 3, we describe the process of creating a new interactive graphics task, from its definition and the requirements on the interactivity to its evaluation. Section 4 presents the stated hypotheses, accompanying experiments and results obtained. The motivation of students is also considered. The Section 5 gives the concluding remarks.

2. Previous studies

Many courses include a diversity of interactive materials in order to demonstrate various concepts, such as algorithms or construction procedures. The inclusion of these interactive materials allows the student to investigate the influence of parameters on the final result and thereby deepens the student's understanding of a presented topic. It is well known that there are several learning styles: sensing vs. intuitive, visual vs. verbal, active vs. reflective, and sequential vs. global [14, 15]. On a well organized course, the learning materials should be appropriate for all learning styles. Based on our experience, many students, especially in the technical sciences, are visual learners, and interactions with animations embedded in learning materials emphasize this trait. Therefore, to better support such learning styles, the idea is to develop learning and knowledge assessment that is more visually attractive and more amusing for the student. Using the proposed gadgets we can achieve that goal. On the other hand, the proposed gadgets can function well as a supplement to the more traditional textbooks and presentations, and can bring balance to engineering instruction that is biased heavily towards reflective, intuitive, verbal, and sequential learning styles [36].

Interactive applications can be very helpful as learning materials. One repository of these applications intended for computer graphics is the interactive tutorial for learning OpenGL [33]. Through this tutorial, parameters of OpenGL functions can be interactively controlled, and their influence on the virtual scene is instantly visible. This provides the students with important feedback and allows them to gain a deeper understanding of the topic. The drawback of these applications is that the users must download the applications' source code and then build the applications. Downloading or running executables from the Internet also presents a security risk. On the other hand, the availability of the source code is very important for better understanding of the concepts presented and for the development of similar applications. Nevertheless, interactive applications that run directly in a web browser under the browser's security constraints, such as Java applets or Flash animations, are more convenient.

The use of interactive graphical applets is especially popular in computer graphics, not only because it is natural to apply acquired knowledge and skills to the learning process itself, but also because some ideas such as: illumination and shading, color modeling, and transformations are difficult or even impossible to explain without figures or interactive tools. The Applet collection [23] that corresponds to one or more figures from Gerald Farin's book [13] is an example in which the difference between simply observing static figures in a book and playing with revived versions of static figures is obvious. Interactivity always offers the possibility of investigation and exploration, which makes it possible to examine special cases that are not obvious from static figures.

The example of teaching "convolution", which is an integral concept of the computer graphics curricula, is presented in [20]. In this study, Hanisch and Straßer defined the concept of "teaching gems" and presented the process of making them. Students first explored the prepared example applet, which presents continuous-time convolution, and then they learned about the transition from the continuous to the discrete domain. Finally, for homework assignment, they had to make their own discrete-time convolution applet. Interactive material is generally difficult to create. However, in the described approach, students were first lead through known examples, then the deconstruction of the topic under consideration to unit tasks was explained to them, and finally they made a new interactive applet. A similar example with quite an impressive collection of the developed interactive graphics applets is presented by Czanner *et al.* [7, 8].

Because the development of interactive learning objects is extraordinarily time consuming and difficult, Laleuf and Spalter decided to create a repository of reusable software components, which should decrease the length of the development cycle [27]. To deconstruct software into reusable components, component granularity and the appropriate level of object complexity for each component need to be determined. However, the existence and use of previously developed components could crucially influence the development of interactive materials and does not require rewriting components from the start. The only drawback is that the development of such components is demanding and time consuming.

All of the aforementioned approaches are focused on the design of interactive teaching and learning materials. In this paper, we propose interactive graphics materials that are suitable for both learning and knowledge assessment. The key point in our approach is the addition of the evaluation of the gadgets. This allows us to address specific questions to the student, allows the student to input the solution, allows us to run an evaluation procedure to determine the correctness of the solution and, finally, the evaluation result and the eventual feedback is presented to the student.

To provide the auto-evaluation ability is in itself a challenging task; however, it is even more challenging if the problem must be represented and solved graphically. The problem generated for a student by a gadget must be carefully crafted. To that end, adequate solution representation is very important; care must be taken to ensure that a solution exists. The evaluation procedure should be developed in such a way as to accept any of the correct solutions provided by a student for a case where the solution is not unique. The benefit of the proposed approach is that evaluation is fast and objective. Additionally, the proposed software environment gives gadgets the ability to track their users and to memorize the previously created tasks for each user. This way each user can reopen any of the previously generated tasks and obtain the correct solutions.

In this paper we also propose using the gadgets in two different ways. Gadgets can be used for learning and knowledge assessment by solving the problems that are generated by the gadgets. However, students can also be involved in the development of the gadgets themselves. In that case, students are responsible for the entire development of the gadget that will be used later for learning by their peers.

In this paper we will focus (without the loss of generality) on the application of gadgets in the teaching of computer graphics. More details and the overall software architecture of the gadget platform can be found in [9].

3. Design process for interactive graphics gadgets

At our institution, through the Interactive Computer Graphics (ICG) course, students are introduced to various computer graphics topics, ranging from graphics hardware, modeling techniques for objects and scenes, raster graphics, mathematical methods and algorithms, interpolation and curves, clipping, culling, illumination, and shading. So we wanted to be able to equip all of these topics with appropriate gadgets that students could use for learning, experimentation, and knowledge assessment.

The main idea of the approach that we propose in this paper is to first introduce students to the concept of graphical gadgets by giving them a homework assignment that is composed of previously developed gadgets. After that, the method for creating gadgets is explained to the students, as well as the expected problems. From the instructor's point of view, the learning objective for each new gadget is stated, and gadget development is deconstructed into one or more elementary tasks. Each task is then analyzed and the presentation strategy and the appropriate level of interactivity are defined and presented to the students. The activities that instructors must take into account are the following: definition of the basic learning objectives, definition of the required functionality of each gadget, and selection of the appropriate evaluation procedures. In most cases, the creation of gadget specifications for a given learning objective is straightforward. However, in some cases, the specification of the gadgets functionality occurs in reverse. In this approach, the problems' solution is first analyzed, and then the specification is created based on the solution. Sometimes, a specific simulator must even be developed to support solution evaluation.

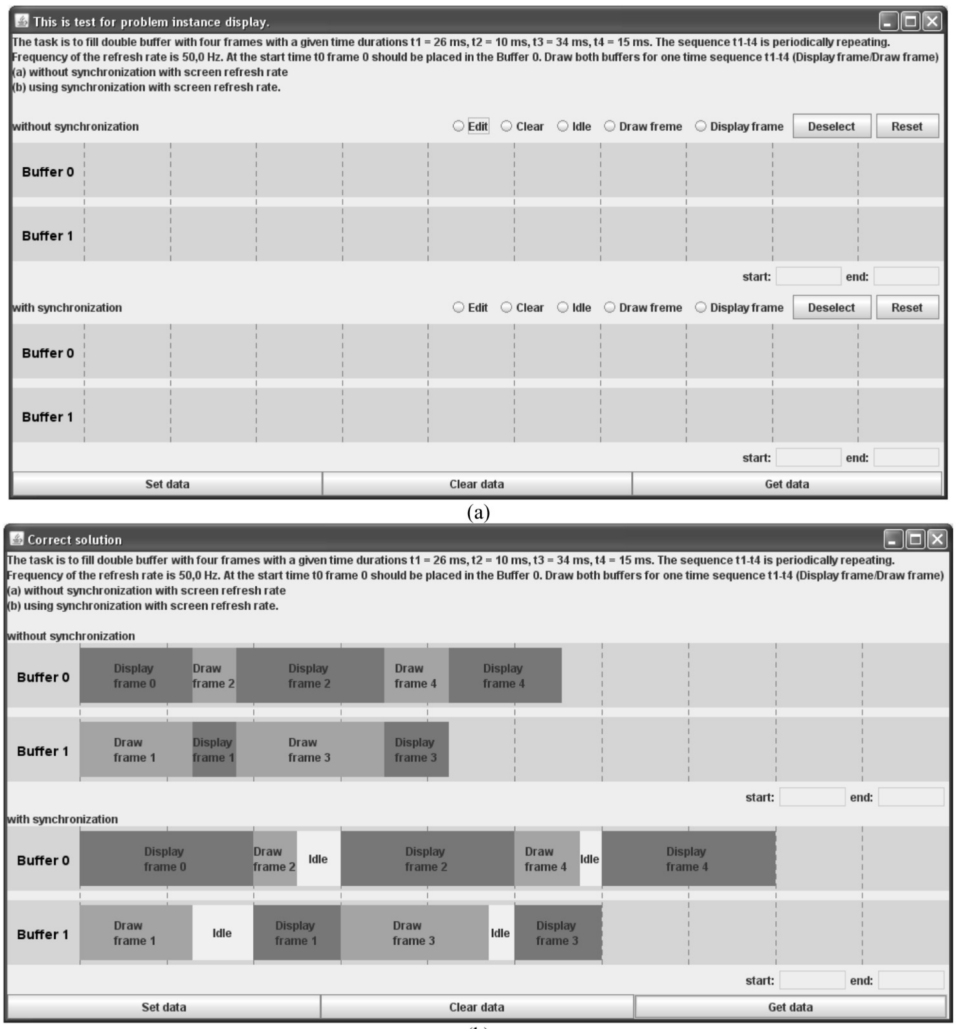
During the definition and development of the gadgets' user interface, students developing the gadgets are instructed to take special care to account for specific cases that may occur and that may produce undesired confusion during the solving process. After the teacher presents and explains these cases to the students who will develop new tasks (tasks, graphical tasks and gadgets from now on will be used interchangeably), the students start the development and programming phase. Through several iterations of testing and improvement, new tasks are created for homework assignments. Students in the same generation test and verify given tasks, and constructive criticism is used to suggest improvements. In that way, students are involved in all aspects of the evolution process, and the developmental cycle is completed.

3.1 Homework assignments

For the last several years, at the Faculty of Electrical Engineering and Computing, University of Zagreb, in the Interactive Computer Graphics course we have had three homework assignments. In each homework assignment, students were given seven to ten tasks. Approximately one week before the mid-term or final exam, the students were given a homework assignment and they had a week to study and solve the tasks. For all of the assigned tasks, students were provided with automatically generated correct solutions after the homework was finished. Students were usually very critical of the various task implementation details, such as the representation of the solution or inappropriateness of the task interactivity. Often, they also offered new ideas for improvement of the noticed imperfections. While there was little progress in the interaction abilities of certain tasks, the key point is that an analytical view was stimulated and the motivation for improvements or even the creation of new materials was achieved.

An example of a task is presented in Fig. 1. The goal of the task is to understand the concept of double buffering. For this process, a student must first understand that when one frame is displayed on the computer screen from one buffer, the next frame is drawn in the other buffer. After drawing is completed in the hidden buffer, the two buffers swap roles. The next step is to synchronize the drawing and swapping process with the screen refresh rate to avoid temporal artifacts.

In this particular example, during the task instantiation the time required to draw each of the four frames is chosen based on random values from the predefined time range. Empty buffers are presented in Fig. 1(a). To solve the task, the learner starts by interactively filling in the two buffers with four frames without synchronizing with the refresh rate, as presented in the first two rows in Fig. 1(b). The next two rows present a situation in which the two buffers are filled in and synchronized with the refresh rate. If the duration of the frame drawing is greater than the synchronization cycle, an idle time is inserted, and the same frame is repeatedly refreshed until the synchronization point is reached. For this task, the ability to graphically solve the task is not mandatory because the task's solution could also be provided by simply entering the time-stamps for each frame. However, the ability to graphically solve the task makes it more attractive, facilitates its usage and is actually a natural way to solve this


This is test for problem instance display.

The task is to fill double buffer with four frames with a given time durations t1 = 26 ms, t2 = 10 ms, t3 = 34 ms, t4 = 15 ms. The sequence t1-t4 is periodically repeating.
Frequency of the refresh rate is 50,0 Hz. At the start time t0 frame 0 should be placed in the Buffer 0. Draw both buffers for one time sequence t1-t4 (Display frame/Draw frame)
(a) without synchronization with screen refresh rate
(b) using synchronization with screen refresh rate.

Fig. 1. The concept of double buffering without and with synchronization is presented. (a) The buffers are initially empty. (b) Four frames are placed in the two buffers, without synchronization in the first two rows and with synchronization in the next two rows.

particular problem. The presented gadget offers an auto-evaluation procedure that also accepts partially correct solutions.

3.2 The design of gadgets

In the learning process, various levels of cognitive skills can be targeted. At the lowest level, recall or recognition of specific facts and concepts in terms of the learned material could be simply tested. In the homework assignments, understanding of the presented concepts is expected from students. In laboratory exercises, the development of various computer applications is expected, which tests the students' understanding of the topic. The creation of new tasks for assignments uses the students' ability to analyze previously developed tasks and to synthesize new ones. We will now consider several of the most important aspects of the gadget development.

3.2.1 Definition of the new tasks

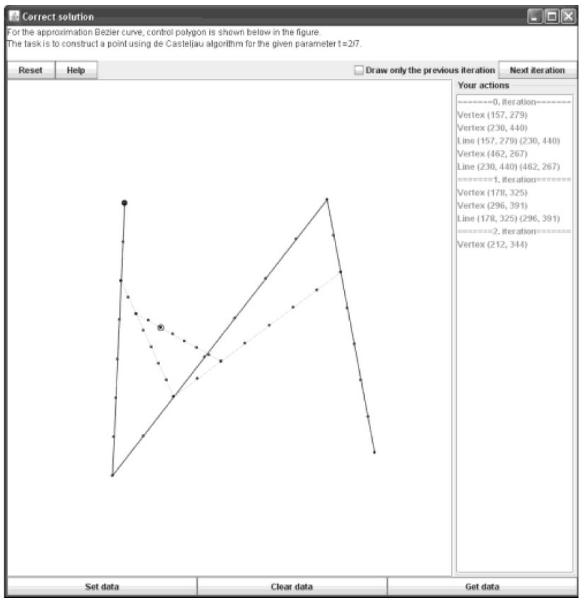
For the creation of new materials, the instructor should specify the lecture units to be appropriately covered. Then, each lecture unit should be deconstructed into elementary units in such a way that each elementary unit still remains interesting and sufficiently motivating, such as, for example, puzzles. For each elementary unit one or more tasks should be created. A task should not be too complicated, but should be challenging enough to grasp a fundamental concept and be intriguing to solve. The main differences between the knowledge assessment applets and learning applets are that the former contain a specific problem that student must correctly solve and that the students have access to evaluation procedures. The latter typically do not entail a particular problem and associated evaluation procedure but enable the student to explore the presented topic. In that sense the gadgets we propose can be categorized as knowledge assessment applets. Performance feedback is crucial to allowing a student to obtain a better picture of his or her understanding of the topic. The need for and the existence of evaluation require more serious consideration than for applets intended only for learning. At the same time, evaluation could be used to motivate competition among students, for example by publishing students' score lists, or to give some prizes [7].

The task's presentation and first impressions are very important to learners. It is undesirable to scare the students with the first appearance of a gadget but it should provoke imagination and curiosity. To achieve that objective, the use of the developer's imagination is desirable.

3.2.2 Interaction

Interaction is another important factor. Interaction should be based on simple mouse clicks or mouse movements as much as possible. It is also important that the interaction is highly intuitive so that the student can become accustomed to it. Inappropriate or tedious interaction will result in frustrated students or even in the abandonment of tasks. In any case, additional instructions should always be provided and be accessible directly from gadgets. The overall simplicity of use is also important, i.e. although additional instructions are desirable, endless instructions must be avoided.

Figure 2 shows an example of a gadget in which the construction of the Bézier curve is required [13]. The de Casteljau algorithm [13, 23] is a recursive method of polynomial evaluation in the Bernstein form. Consecutive linear interpolation with a given parameter leads to the final point on the curve. In this example, the parameter *t* and a control polygon are given in advance (in the particular example, *t* = 2/7). To solve the task, the student must divide each segment of the polygon into the required number of sub-segments (in accordance with parameter *t*) by right-clicking on each of the segments the appropriate number of times. Then, the resulting division points must be appropriately interconnected, resulting in a polygon with one less segment. On the new polygon, the described procedure is repeated until the resulting polygon is reduced to a single segment. Division of that segment in accordance with parameter *t* results in the final point, which belongs to the Bézeir curve. In Fig. 2, the segments of a three-lined polygon are divided with parameter *t* = 2/7 and connected, resulting in a new two-lined polygon. In the next iteration, segments of the obtained two-lined polygon are further divided and connected, resulting in a single-lined polygon. Its division then produces the final point of the curve.



For the approximation Bezier curve, control polygon is shown below in the figure.
The task is to construct a point using de Casteljau algorithm for the given parameter t=2/7.

```
=======0. iteration=======
Vertex (157, 279)
Vertex (230, 440)
Line (157, 279) (230, 440)
Vertex (462, 267)
Line (230, 440) (462, 267)
=======1. iteration=======
Vertex (178, 325)
Vertex (296, 391)
Line (178, 325) (296, 391)
=======2. iteration=======
Vertex (212, 344)
```

Fig. 2. Interactive construction of the Bézier curve (de Casteljau's algorithm).

3.2.3 Evaluation

Evaluation of task solutions can be a challenging problem. Solutions that are produced by students can be correct, partially correct, or incorrect. In an attempt to avoid binary grading (correct or incorrect), partially correct solutions should have an associated correctness measure. For example, in Fig. 1, if the first two frames are correctly placed into a double buffer, or in Fig. 2, if the first iteration is correct, the evaluation procedure should produce a value that corresponds to a partial amount of correctness. This often proves to be rather problematic. To illustrate, we will focus on a specific example depicted in Fig. 2 and assume that the student only created a simpler two-lined polygon. We can reason that to correctly solve the problem, the student must create a total of six division points and draw a total of three line segments. That represents nine units of work. If the student only created a new two-lined polygon (i.e., created three division points and then connected two lines), that would make five units of work. The correctness measure can then be defined as $5/9 \approx 0.55$. However, it is important to note that there is no unique way to calculate this measure. For example, it is not possible to evaluate a solution in which the student created a two-lined polygon but from incorrect division points or a solution in which the first division is incorrect but the "procedure" is correct. It is also important to note that the approach to base grading on the specific sequencing of the student actions is just one among many. Depending on the task at hand, different evaluation procedures may be more appropriate and are supported by our framework. There are many considerations that should be accounted for in the evaluation procedure; in addition, it would be ideal if the evaluation procedure could provide a sentence or two as feedback to student that explains what was wrong and how to correct it.

In Fig. 3, the task is to build a Binary Space Partitioning (BSP) tree [18] for a scene depicted at the left side of the figure. Each of the edges in the scene can be chosen and placed to create a node in the BSP tree on the right side of the figure. In this example, there are many possible correct solutions, which depend on the order in which edges are chosen and placed in the tree. The evaluation procedure must evaluate all possible solutions and accept all correct ones. Here, the partially correct solutions complicate the evaluation process even more. Hence, for this task, the evaluation procedure is particularly complex.

The task presented in Fig. 3 is freely available on the Internet [32]. Students can practice building BSP trees or creating sorted lists of polygons for a given randomly created tree at any time. Repeated playing with the given tasks leads to more in-depth learning. For any randomly created task, a measure of correctness is generated for the student's solution, so that they can learn through practice.

When the students are first confronted with new materials, it is desirable for the accompanying tasks to present a main concept but also to be as simple as possible. As learning progresses, more complicated tasks are appropriate. Therefore, it is desirable that the level of complexity for the tasks created by gadgets can be adjusted. In the example depicted in Fig. 3, students can adjust the number of objects that will be presented on the scene and their type (lines, triangles, or squares), thereby adjusting the level of complexity.

3.2.4 Special cases

Particular care should be given during the gadget development to special cases that could occur. For example, when a triangle is generated by randomly selecting coordinates for its three vertices, a degenerative case can occur, in which all three vertices lie on the same line (the triangle collapses into a zero-height triangle). The occurrence of degenerative cases must be prevented during the creation of the scene. Some degenerative cases are predictable and manageable, but others are very hard to predict and appear only when they are least expected or least desirable. Only experience helps to prevent the appearance of undesired cases. Therefore, it is the instructor's responsibility to accumulate knowledge about potential special cases that may occur. The instructor should warn the students about these special cases before assigning the task to develop a specific gadget to each student. It is then the students' responsibility to create a gadget in such a way as to ensure that described degenerate cases do not occur during the problem instantiation or to implement sufficient validation procedures, so that generated problems are free from such cases.

Another problem is when the solution input requires more screen space than is available. During the creation of an individualized task, the procedure must be aware of the constraints in which the problem will be solved and ensure that the correct solution can be produced by the student under those constraints.

Because the creation procedure is driven by randomness, it is possible that the procedure fails to produce a valid task. In that case, it is recommended that the creation procedure is repeated until a valid task is generated or until a predefined number of attempts is reached. In the latter case,

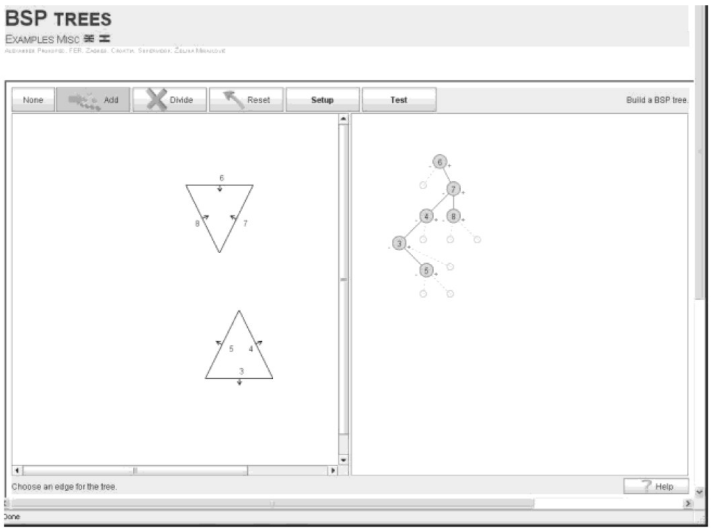


Fig. 3. The task depicted is to create a BSP tree (right) for a given scene (left).

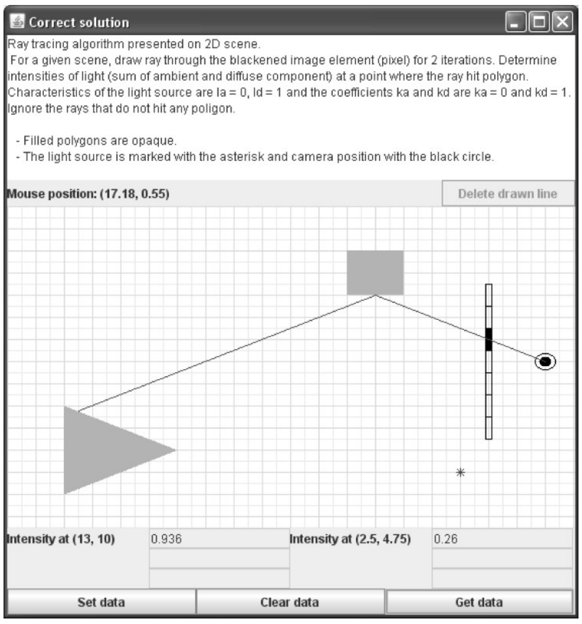
the creation procedure should return an example of a task prepared by the developer, and that example task should be used to assess the student's knowledge (as a fail-safe task).

3.2.5 Combining interactive and numerical solutions

In some cases, it is appropriate to combine interactive graphical solutions with numerical results. For example, in Fig. 4, the concept of ray tracing is presented. The scene consists of two triangles, the light position (marked as *), the eye position (a black double circle), and one marked pixel in the screen space. This scene is a 2D representation or a cross-section of a 3D scene. The student's task is to find the ray from the eye point through the marked picture element and the intersection of the ray with the triangles in the scene. According to the eye and light positions and other given parameters, the intensity is calculated for the intersection point and entered as a number. The next step is to calculate the intensity of the reflected ray and accompanying intensity of the point at which the reflected ray intersects the triangles, if that point is visible from the light source. In this example, the most appropriate solution is to interactively set the ray and numerically write the calculated intensity in the input box.

3.2.6 Preparing students for the implementation of tasks

Before the students start to implement the required tasks, all of the requirements and problems that might appear need to be explained to them. All of


Correct solution

Ray tracing algorithm presented on 2D scene.
For a given scene, draw ray through the blackened image element (pixel) for 2 iterations. Determine intensities of light (sum of ambient and diffuse component) at a point where the ray hit polygon.
Characteristics of the light source are Ia = 0, Id = 1 and the coefficients ka and kd are ka = 0 and kd = 1.
Ignore the rays that do not hit any poligon.

- Filled polygons are opaque.
- The light source is marked with the asterisk and camera position with the black circle.

Mouse position: (17.18, 0.55)

Intensity at (13, 10) 0.936
Intensity at (2.5, 4.75) 0.26

Fig. 4. Interactive graphical solution is combined with a numerical value that represents the intensity of light at the observed point.

the tasks will be created using Java programming language. Java is a freely available object-oriented programming language that offers exceptional portability. Applications that are written in Java work on almost any modern operating system, including desktop computers, laptops, and mobile devices. The language is especially appealing because it works in web browsers, which makes it independent of the actual operating system. Prior to their enrollment in the Interactive Computer Graphics course, we offered the students a chance to learn the Java programming language in the form of a separate course: Introduction to the Java programming language.

Before students begin to develop new tasks, they are introduced to the structure of a task. Each task is composed of three components: the instantiator, the presenter, and the evaluator. The instantiator is the component that creates new parameterized task instances; this allows the creation of an individualized task for each student. The presenter is the component that is responsible for task presentation. For interactive graphical tasks, the presenter takes the form of an applet and enables the student to see and solve the task. The evaluator is a component that is responsible for the evaluation of the student's solution and the creation of the correctness measure and feedback message. Developing everything in Java enabled us to offer to students a uniform programming environment for the development of all the task components.

For each student who decides to participate in the creation of a new task, an account is created in the svn (http://subversion.apache.org/) repository. This repository contains all of the previously developed problems and provides access to various build scripts and codes. It is also useful for students to learn from previously developed tasks. The code is built using the Apache Ant (http://ant.apache.org/) build system. Although the actual coding can be done using any integrated development environment (IDE) for Java, we encouraged the students to use the freely available Eclipse IDE (http://www.eclipse.org/), for which we could offer them support. During the development of tasks, students are offered assistance, either with Java-specific problems or computer graphics concepts. Students can also collaborate during the development of tasks; this often happens during lectures or later by means of various forums and e-mail exchange.

Once the students have finished developing the given tasks, a presentation of tasks is organized to debug them and further improve their quality. The final verification of the developed tasks is their inclusion in the next homework assignment, which is then given to every student enrolled in the course. Nevertheless, we always organize the meeting with students where they can present and demonstrate their solutions and if it is correct we accept it. If necessary, we organize corrections and improvements of all previously developed similar gadgets. Once this procedure is completed and all bugs are corrected we can rely on the developed gadget. This final test often provokes a number of comments and suggestions on further improvements to the tasks' quality. At the same time, the tasks can instantly be used by other students to learn the underlying computer graphics concepts. This instructive thorny path brings us back to the beginning, but we gain new experience, and the library of tasks is increased and prepared for the next generation of students.

4. Considered hypotheses and results

The usage of the proposed software environment for learning and knowledge assessment was introduced in the Computer Graphics (CG) course in academic year 2006/2007. That year the challenge to develop new gadgets was accepted by five students. In the academic year 2007/2008, due to curriculum change, the Computer Graphics course was held for the last time and the Interactive Computer Graphics course was introduced. That year, three students were involved in gadget development. The number of students grew to fifteen in the third generation and four students were involved in the next two academic years. Table 1 summarizes the number of students enrolled in the Interactive Computer Graphics (ICG) course for each academic year, the number of students involved in task design, and the total number of tasks developed. The differences in the number of students correspond to the transition period of the Bologna process at our university. The number of students involved in the development of new tasks oscillated, but the trend shows a constant interest in participation in the creation of learning materials.

Not all of the developed tasks were interactive, but nearly half of them were. The other half were parameterized tasks in which certain calculations were required to complete the task and the solutions were input numerically. Examples of these non-interactive, parameterized tasks include: finding the minimum distance between two straight lines where the line coefficients are varied, and finding the distance between a point and a plane, in which the coordinates of the test point and coefficients of the plane are varied. With the joint effort of teachers and students participating in gadget development, over the five academic years we compiled a total of 58 gadgets, which are regularly used for learning and knowledge assessment. We named the developed task in the form of gadgets *GadgetTask*.

Starting from the academic year 2007/2008, gadgets were introduced to all students through several homework assignments as a part of the preparations for course exams. However, since we only conducted detailed mid-term and final exam analyses from the year 2008/2009, we will provide the data from these academic years only. For the purpose of our experiment, we created two independent divisions of all tasks from mid-terms and final exams. In one division, which is represented as columns in Table 2, the first category comprised exam tasks for which similar ones were provided in the homework assignments (denoted as *GadgetTask.InExam*), which preceded the exam. It is important to clarify the term "similar". We do not consider it to mean "almost the same". We considered it as if the student successfully solved the task that is based, for example, on translation and rotation, she/he will be able to apply that knowledge in any other context involving translation and rotation. The other category contained tasks for which the similar ones were not given in the homework assignments but were explained in the classroom or laboratory (denoted as *NonGadgetTask.InExam* in Table 2). The *GadgetTask.InExam* were further divided into two additional groups: *GadgetTextualTask.InExam*, for which the similar parameterized task in a textual form was provided in homework assignments, and *GadgetGraphicalTask.InExam*, for which the similar interactive graphical parameterized tasks were provided in homework assignments. In the second division of exam tasks, which is represented as rows in Table 2, the first category (denoted A) contained multiple-choice questions: tasks in which the student was provided with possible answers and had to choose the correct one. In the other category

Table 1. Summary of the number of gadgets developed in each academic year

Academic year	Number of students enrolled on course	Number of students participating in applet design	Total number of applets developed
CG2006/07	57	5	20
CG + ICG 2007/08	65 + 101	3	25
ICG2008/09	136	15	50
ICG2009/10	103	2	54
ICG2010/11	90	2	58

Table 2. Measures of success for different task categories

Academic year and task category	**Number of students taking exams**	***NonGadgetTask.InExam* (success measure)**	***GadgetTask.InExam* (success measure)**	
08/09, A	118	0.633	0.656	0.658
09/10, A	89	0.516	0.612	0.686
10/11, A	68	0.391	0.631	0.765
08/09 & 09/10 & 10/11 B	118 + 89 + 68	0.476	0.477	0.742

(denoted B), questions were open-ended, so a complete calculation procedure was inspected and evaluated by a human grader.

The values in Table 2 present measures of success achieved by students for the exam tasks in each category and academic year. For each task solved by student, a correctness measure was calculated, ranging from 0—incorrect to 1—correct. Values in between represent partially correct solutions. The success measure for each category was then calculated as the average of the correctness measures of tasks from that category. The data provided in Table 2 include only the students who attended all three exams (e.g., Table 1 shows 136 students on ICG2008/09 but Table 2 shows, for the same academic year, only 118 students). For category B the average for all years is presented because only four open-ended tasks for each homework were given, so the sample size is relatively small for further categorization in *GadgetTask* categories for each year. We believe that some deviations in the results can be expected and we think that this is due to the statistical fluctuation. Hence, we will focus our attention on the general trend of indicators in the main categories.

The first hypothesis we wanted to investigate was that the usage of tasks provided by gadgets lead to similar or better learning results than traditional methods. Our assumption was that students will perform better on mid-term exams tasks for which similar tasks had been included as gadgets in homework than on mid-term exam tasks for which similar ones were explained in the classroom or in laboratory. As can be seen from Table 2, students achieved better results for tasks from the *GadgetTask.In Exam* category than for the tasks from the *NonGadgetTask.InExam* category. From this, we conclude that the homework that comprised individualized tasks helped students to learn and accept the course material. When comparing the tasks from the *GadgetGraphicalTask.InExam* and *GadgetTextualTask.InExam* categories, we also observed better student achievements when solving mid-term exam tasks similar to *GadgetGraphicalTask.InExam*. This is a clear indication that interactive graphical tasks can additionally foster the learning process and help students gain a deeper understanding of the material.

The next hypothesis that we wanted to investigate was that the interactive graphical gadgets are more attractive to students than textual ones. In the experiment, we compared the number of solved and unsolved tasks as well as the measures of success for the interactive graphical tasks and the tasks that are not interactive. Table 3 presents the results. The total number of task instances in three homework assignments in academic year 2008/09 was 2414. Of these, 1202 were *GadgetTextualTask* and 1212 were *GadgetGraphicalTask*. Each homework assignment contained seven tasks. Homework assignments are not obligatory, but five credit points are offered for correctly solved homework assignments. In Table 3, the column labeled *Total number* contains the total number of task instances created. The column labeled *Solved* gives the number of task instances that were solved by students (correctly or incorrectly). The column labeled *Not Solved* gives the number of task instances that the students did not try to solve. Finally, the values in the column labeled *Success measure* were calculated as in Table 2, using only the solved task instances.

As seen from Table 3, the solvability of the interactive tasks was better. Therefore we may conclude that more students were attracted to interactive graphical tasks than to the *GadgetTextualTask*-ones. From the same table we also see that

Table 3. Comparison between the *GadgetTextualTask* and the *GadgetGraphicalTask* tasks

Academic year	**Tasks**	**Total number**	**Solved**	**Not solved**	**Success measure**
2008/09	*GadgetTextualTask*	1202	1099 (91.4%)	103 (8.6%)	0.689
	GadgetGraphicalTask	1212	1134 (93.6%)	78 (6.4%)	0.799
2009/10	*GadgetTextualTask*	768	700 (91.1%)	68 (8.9%)	0.772
	GadgetGraphicalTask	959	928 (97.3%)	26 (2.7%)	0.845
2010/11	*GadgetTextualTask*	641	593 (92.5%)	48 (7.5%)	0.819
	GadgetGraphicalTask	856	820 (95.8%)	36 (4.2%)	0.854

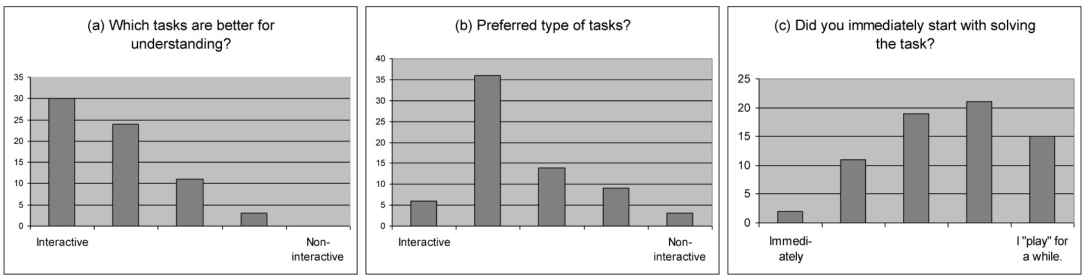


Fig. 5. Students' opinions about the interactive graphical tasks.

the success measure for the interactive tasks was better than that for non-interactive tasks. From this, we may assume that interactive graphical tasks allowed students to experiment with the task, and so to better understand it and solved it.

To further investigate and justify these observations, we conducted a student poll using a written questionnaire. The students were asked three questions about their preferences concerning the type of gadgets and the influence of the gadgets on their understanding of the material. The questions and results obtained are given in Fig. 5. The questions had 5-grade Likert scale. The students' opinion indicated that they preferred interactive graphical tasks, as presented in Fig. 5(b). Additionally, it was the students' opinion that interactive graphical tasks are better for understanding (Fig. 5(a)) and exploration (Fig. 5(c)) of the presented topics. In addition, when the students were presented with interactive graphical tasks that had intuitive user interfaces, they often started playing with the task to explore its capabilities and to observe the results. This was perceived as fun and similar to playing a game.

The third hypothesis we wanted to investigate was that the homework using gadgets can have a positive impact on students' performance on the mid-term exam that follows the homework. To investigate this, we evaluated the relationship between the students' success on homework assignments (HW1, HW2, and HW3) and on the midterms (MT1 and MT2) and final exam (FI) using linear regression analysis.

In the regression analysis, we included only the students who completed both the homework assignment and the related exam (presented as the number of observations in Table 4). The sample on which the analysis was performed is rather large, and the Pearson's correlation coefficient has a moderate positive value, clearly indicating some positive relationship. The analysis using Pearson's correlation coefficient showed a statistically significant linear relationship at p = 5%.

4.1 *Motivation of students*

Our opinion is that the development of new interactive graphical tasks should not be mandatory for all students in introductory courses, but it is intriguing and challenging for some of them. Consequently, in the Interactive Computer Graphics course, we offered five extra credit points to students who were willing to attempt the development of new tasks. In addition, there is a possibility that students have to solve the tasks developed by themselves. We considered this to be a bonus for students who are included in the development task. On the one hand, they were proud to be involved in the creation of new materials and to distinguish themselves among their colleagues; on the other hand, they were aware of their responsibility. They were also simultaneously subject to criticism and compliments from their peers. The students were aware that this is an opportunity to develop software that will be used in practice. These students moved from being the consumers of the assignment tasks to being the

Table 4. Linear regression analysis exam results and corresponding activity in homework

Predict.	**Num. of observations**	***R***	***P* value**	***t*-stat**
HW1-MT1 08/09	117	0.264	3.97E-03	2.94
HW2-MT2 08/09	114	0.287	1.88E-03	3.18
HW3-FI 08/09	110	0.231	1.46E-02	2.48
HW1-MT1 09/10	89	0.468	1.96E-06	5.08
HW2-MT2 09/10	82	0.429	5.82E-05	4.25
HW3-FI 09/10	76	0.352	1.84E-03	3.23
HW1-MT1 10/11	71	0.447	9.23E-05	4.15
HW2-MT2 10/11	69	0.451	1.27E-04	4.07
HW3-FI 10/11	69	0.357	2.54E-03	3.13

developers. This is a very important transition because we have to prepare the students for real-world tasks and the accompanying responsibilities through graduate education.

The new generations of students are immersed in technologies and an abundance of information, and they are extremely willing to support and participate in the development of any materials that are based on new technologies. This leads to the development and usage of novel approaches in education. Students are now "digital natives". They are familiar with social networks and accustomed to constant communication, multitasking, chatting, involvement and participation. As "digital natives", it is very important to them to participate and to contribute to the creation of learning materials because it is simply in their nature. The instructor's responsibility is to provide them with the opportunity to do so.

By supporting new approaches and technologies in education, we adapt learning materials to the students' ways of thinking and to the demands of new generations, making the course materials sustainable. Problem-based learning, the obvious applicability of the knowledge gained and the involvement of students in the creation of course material are the most important characteristics of the approach described. Learning by solving concrete problems helps students see the importance of the subject, enables them to retain the acquired knowledge for longer, and allows them to recognize and solve similar problems. By developing new tasks, the students are involved in a small part of a rather large project and have the opportunity to experience working in a team. As future engineers who will shape our future, it is important for them to be aware of their own impact.

The approach described in this paper was exemplified in an Interactive Computer Graphics course. However, it is important to emphasize that the approach described is also applicable to a variety of other courses. For example, interactivity is appreciated in physics, where it can allow students to experiment and simulate various physical behaviors and interactions; in mathematics, where it can be used to visualize the meaning of derivations; and in signal processing, where it can be used to visualize and explain the DFT (Discrete Fourier Transform). We expect that students who attended the Interactive Computer Graphics course and developed one or more gadgets will also be willing and able to develop similar gadgets for other courses and have gained some benefit on those courses as well. In our faculty we also have some successful implementations for the Artificial intelligence and Digital logic courses.

5. Conclusions

The main objective of this paper was to propose new concepts for knowledge assessment. The proposed development of knowledge assessment materials is rather complex. However, the final result is then reusable because parameterization is included. Knowledge assessment materials could easily be categorized and used in the future for adaptive and personalized knowledge assessment. Another benefit is the participation of students in course material development and in the creation of useful products.

As the conducted survey indicated, the developed problems helped the students to gain a deeper understanding of the subject matter. This is particularly true in the case of interactive graphical tasks. These graphical tasks showed two additional benefits. First, the students liked them more than the pure textual tasks. Second, usable interactive graphical tasks were often played with and were the subject of investigation and exploration—a process that further helped the students better understand the subject matter.

For the future, we plan to prepare a collection of all the developed tasks and allow students to explore and practice each task as much as they want and at their own pace. We would like to investigate how often and which tasks students like to practice. For additional motivation, we plan to allow the students to use pseudonyms and participate in a "learning game", for which a top-score list will be maintained. In this way, we hope to stimulate a competitive spirit among the students and make learning by practicing more intriguing. The limitations of the proposed work are that it is not appropriate for essay type assessment. For that type of assessment, text mining techniques are required as well as the semantic analysis of the text. That is out of the scope of the proposed paper.

Combining elementary tasks into more complex ones can further increase task intricacy. To solve them, students will then have to engage their higher cognitive skills. Because solving these more complex tasks requires the combination of knowledge fragments, it emphasizes more sophisticated goals and strategies. The idea of this approach is to obtain more contextualized information because having a clear context while learning helps to retain the learned material. We also plan to provide the students with several different ways of learning a certain topic and to track which of them is preferable. For example, when a more complex task composed of several elementary units is put forward, we plan to offer the students all of the necessary links to materials that cover each unit. Furthermore, for each unit, we plan to offer more than one type of material: textbook chapters, applets, and interactive tasks. The final goal is to obtain a better insight into the type of materials that students prefer.

In conclusion, the developed graphical tasks could influence the curricula. The course curriculum and program structure should provide students with the key elements of both theoretical and applied knowledge that is required to adequately address the practical problems. However, involvement of students in gadget development is actually both problem-based learning and a personalized approach. Our very positive experience in that aspect indicates that problem-based learning and a personalized approach should both be applied to the curriculum.

References

1. R. Abiyev, D. Ibrahim and B. Erin, EDURobot: an educational computer simulation program for navigation of mobile robots in the presence of obstacles, *International Journal of Engineering Education*, **26**(1), 2010, pp. 18–29.
2. The BlackBoard Learning System, http://www.blackboard.com, July 2011.
3. I. Boticki, C.-K. Looi and Wong, L.-H., Supporting mobile collaborative activities through scaffolded flexible grouping, *Educational Technology & Society*, **14**(3), 2011, pp. 190–202,
4. P. Brusilovsky and S. Sosnovsky, Individualized exercises for self-assessment of programming knowledge: An evaluation of QuizPACK, *ACM Journal on Educational Resources in Computing*, **5**(3), 2005, http://doi.acm.org/10.1145/1163405.1163411.
5. P. Brusilovsky and S. Sosnovsky, Engaging students to work with self-assessment questions: A study of two approaches. In *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education,* 2005, pp. 251–255,
6. I. Cheng and A. Basu, Interactive graphics for computer adaptive testing, *Computer Graphics Forum 28*, 2009, pp. 2033–2045.
7. S. Czanner, A. Ferko and J. Stugel, Computer graphics virtual textbook. In *Proceedings of the 24th Spring Conference on Computer Graphics (SCCG '08)*, ACM, New York, NY, USA, 2008, pp. 127–133, http://dl.acm.org/citation.cfm?doid=1921264.1921291.
8. S. Czanner, A. Ferko, J. Stugel, P. Nunukova, Applet competition as an educational tool in creating novel e-textbook, EG 2009, *Eurographics 2009: Computer Graphics Education Workshop*, Munich, 2009, 30.3–3.4.
9. M. Cupic and Ž. Mihajlovic, Computer-based knowledge, self-assessment and training, *International Journal of Engineering Education*, **26**(1), 2010, pp. 111–125.
10. P. Dolog, B. Simon, W. Nejdl, and T. Klobucar, Personalizing access to learning networks, *ACM Trans. Internet Technol.,* **8**(2), 2008, pp. 1–21, http://doi.acm.org/10.1145/1323651.1323654.
11. E. Duval and K. Verbert, On the role of technical standards for learning technologies, *IEEE Trans. Learn. Technol.,* **1**(4), 2008, pp. 229–234, http://dx.doi.org/10.1109/TLT.2009.6.
12. M. Ebner and A. Holzinger, Successful implementation of user-centered game based learning in higher education: An example from civil engineering, *Computers and Education,* **49**(3), 2007, pp. 873–890.
13. G. Farin, *Curves and Surfaces for CAGD:* A Practical Guide, 5th edn, Morgan Kaufmann Publishers Inc., 2002.
14. R. M. Felder and L. K. Silverman, Learning and teaching styles in engineering education, *Engineering Education*, **78**(7), 1988, pp. 674–681.
15. R. Felder and J. Spurlin, Applications, reliability and validity of the index of learning styles, *International Journal of Engineering Education*, **21**(1), 2004, pp. 103–112.
16. F. Ganovelli and M. Corsini, eNVyMyCar: a multi-player car racing game for teaching Computer Graphics, *Computer Graphics Forum*, **28**(8), 2009, pp. 2025–2032.
17. V. Glavinic, M. Cupic and S. Groš, StudTest—a platform supporting complex and interactive knowledge assessment, *International Conference ICL—Interactive Computer Aided Learning*, Villach, 2008.
18. D. Hearn, and M. P. Baker, *Computer Graphics with OpenGL*, 3rd edn, 2004.
19. N. Hoic-Bozic, V. Mornar and I. Boticki, A blended learning approach to course design and implementation, *IEEE Transactions on Education*, **52**(1), 2009, pp. 19–30 http://dx.doi.org/10.1109/TE.2007.914945.
20. F. Hanisch and W. Straßer, Making of an interactive teaching gem, *ACM SIGGRAPH 2006 Educators Program, SIGGRAPH '06*, 2006, art. no. 1179349,
21. IMS Global Learning Consortium Inc, *IMS specifications*, 2011. Available from: http://www.imsglobal.org/specifications.html
22. C. C. Ko, B. M. Chen, J. Chen, J. Zhang and K. C Tan, A web-based laboratory on control of a two-degrees-of-freedom helicopter, *International Journal of Engineering Education*, **21**(6), 2005, pp. 1017–1030.
23. M. Kraus, Applet collection that corresponds to one or more figures from Gerald Farin's book [13], http://www.vis.uni-stuttgart.de/~kraus/LiveGraphics3D/cagd/
24. M. Lande and L. Leifer, Difficulties student engineers face designing the future, *International Journal of Engineering Education*, **26**(2), 2010, pp. 271–277.
25. D. Lowe, S. Murray, E. Lindsay and D. Liu, Evolving remote laboratory architectures to leverage emerging internet technologies, *IEEE Trans. Learn. Technol.,* **2**(4), 2009, pp. 289–294, http://dx.doi.org/10.1109/TLT.2009.33.
26. IEEE Learning Technology Standards Committee IEEE standard for Learning Object Metadata, *IEEE Standard 1484.12.1*, June 2002, 1484.12.1. Available at http://ltsc.ieee.org/wg12.
27. J. R. Laleuf, and A. M. Spalter, A component repository for learning objects: A progress report. In *Proceedings of First ACM/IEEE-CS Joint Conference on Digital Libraries*, 2001, pp. 33–40, http://doi.acm.org/10.1145/379437.379444.
28. IEEE Learning Technology Standards Committee, 2008, http://ieeeltsc.org.
29. E. Marti, D. Gil and C. Julia, A PBL experience in the teaching of computer graphics, *Computer Graphics Forum*, **25**(1), 2006, pp. 95–103.
30. Available at http://www.moodle.org, July 2011.
31. IEEE P1484.2.5/D8, Draft Standard for Learning Technology—Public and Private Information (PAPI) for Learners (PAPI Learner), 2002.
32. A. Prokopec and Z. Mihajlovic, *Binary Space Partitioning Applet, Interactive Applet for Learning how to Create and use BSP Trees*, 2009, Available at: http://www.zemris.fer.hr/predmeti/irg/BSP
33. N. Robins, OpenGL tutorial, http://www.xmission.com/~nate/tutors.html.
34. Juan Carlos G. de Sande Computer-based training tool for signals and systems exercises, *International Journal of Engineering Education*, **27**(5), 2011, pp. 1150–1157.
35. S. Sheppard, A. Colby, K. Macatangay and W. Sullivan, What is engineering practice? *International Journal of Engineering Education*, **22**(3), 2006, pp. 429–438.
36. L. Thomas, M. Ratcliffe, J. Woodbury and E. Jarman, Learning styles and performance in the introductory programming sequence, *SIGCSE Bull.*, **34**(1), 2002, pp. 33–37. http://dx.doi.org/10.1145/563517.563352.
37. Web3D Consortium—VRML97 and Related Specifications, http://www.web3d.org/x3d/specifications/vrml.
38. Extensible 3D (X3D). Open Standard for Real-Time 3D Communication, http://www.web3d.org/x3d, 2011.

Zeljka Mihajlovic received her B.Sc., M.Sc. and Ph.D. degrees in computer science from the Faculty of Electrical Engineering, University of Zagreb in 1988, 1993 and 1998, respectively. Since graduation, she has worked as a Teaching and Research Assistant with the Department of Electronics at the University of Zagreb, Faculty of Electrical Engineering and Computing, Croatia. She has collaborated on several projects funded by the Ministry of Science, Education and Sports of the Republic of Croatia. Currently, she is Associate Professor with the Department of Electronics, Microelectronics, Computer and Intelligent Systems at the University of Zagreb, Faculty of Electrical Engineering and Computing. She participates in teaching several undergraduate and graduate courses, including Interactive Computer Graphics, Computer Graphics, Computer Animation, Digital Logics, and Visualization. Her primary research interests are in the fields of computer graphics, visualization, volume rendering and interpolation, as well as engineering education and e-learning technologies.

Marko Cupic received his B.Sc. degree in Computer Science in 2002, his M.Sc. degree in Computer Science in 2006 and his Ph.D. in 2011 from the Faculty of Electrical Engineering and Computing, University of Zagreb. He is currently a Researcher at the Department of Electronics, Microelectronics, Computer and Intelligent Systems, of the University of Zagreb, Croatia. His current research interests include soft computation and e-learning.