An Empirical Study on Factors Influencing the Effectiveness of Algorithm Visualization

VASSILIOS LAZARIDIS[1], NIKOLAOS SAMARAS[1], ANGELO SIFALERAS[2]

[1]*Department of Applied Informatics, University of Macedonia, 156 Egnatias Str., 54006 Thessaloniki, Greece*

[2]*Department of Technology Management, University of Macedonia, Loggou-Tourpali, 59200 Naoussa, Greece*

ABSTRACT: The determination of the factors influencing the effectiveness of algorithm visualization poses an interesting research question. In this paper, we present the results of a longitude empirical study regarding this question. The study was based on an evaluation of the Visual LinProg educational tool inside classrooms. Visual LinProg is a web-based educational tool, which solves linear programming problems using animation and visualization techniques. Visual LinProg was developed to be used in linear programming courses to supplement the teaching. Our empirical study is based on questionnaires that include quantitative and qualitative topics. This evaluation first indicates that Visual LinProg facilitates the learning of the Revised Simplex algorithm and second presents more results on factors influencing the understanding of this algorithm by the students/users of the Visual LinProg.

Keywords: educational software, algorithm visualization, linear optimization, empirical studies.

---

1. INTRODUCTION

Algorithm visualization (AV) or, in other words, the dynamic presentation of an algorithm represents a high-level characterization of part of a computer program with a depiction. These visual descriptions are not related to the source code of any program but instead with a high level where only the suitable opinions are presented and the details are waived. AV is based on the basic graphics capability of a home personal computer to demonstrate the algorithmic functionality using animation. A detailed review of AV has been presented by Stasko & Hundhausen in [1]. AV systems offer certain advantages in the educational process, e.g., students can obtain a deeper understanding of the subject being taught, rather than abstract theoretical knowledge. Several studies on AV effectiveness have been reported in the literature (see [2], for a

Correspondence to A. Sifaleras (sifalera@uom.gr).

review), usability evaluations of AV software [3], and empirical studies (e.g., [4, 5, 6]) of educational activities using AV technology.

During the learning process of optimization methods and operational research in general, the study of examples has proved to be effective for the connection of theory with the algorithms and their application for the solution of a problem. The mathematical formulation, the selection of the most suitable algorithm for the solution, and the interpretation of results compose the necessary steps of all optimization problems. Simplex type algorithms are the most common tool for the solution of linear problems (LPs). However, the learning of simplex type algorithms for the solution of an LP faces many difficulties. The major difficulty of a course entitled "Linear Optimization Algorithms" is that the students should be familiar with the basic notions and operations of linear algebra.

Nowadays, an interesting research question is related to the determination of the factors influencing the effectiveness of AV. In this paper, we present the results of an empirical study regarding the above question. The study was based on an evaluation of the Visual LinProg educational tool [7]. It should be mentioned that, to the best of our knowledge, no other educational software exists for teaching the revised simplex algorithm. Therefore, it is not possible to compare our tool against other similar software.

The effectiveness of algorithms' visualization is determined by measuring the pedagogical value [2, 8, 9]. To continue, the pedagogical value of algorithms' visualization describes how many students learned the functionality of the algorithm, which is presented by animation and visualization. The measurement of the effectiveness of visualization often involves three groups of students, e.g., a group using visualization, a group using only a textual description, and a group using a combination of visualization and textual description in order to comprehend the functionality of an algorithm for a given period. Past empirical AV studies [10, 11] are based on the theory of active student participation in the process of visualization. Ten of the fourteen experiments showed that those participants who were actively involved with AV applications learned significantly more than those who passively attended the visualization.

The rest of the paper is organized as follows. Following this introduction, a short description of Visual LinProg is given in Section 2. The research methodology covering the data collection, the synthesis of the participants, and the questionnaire is analytically described in Section 3. Section 4 presents the results of the empirical study and provides a discussion on those topics. Finally, conclusions and future work are presented in Section 5.

2. A SHORT DESCRIPTION OF THE VISUAL LINPROG SOFTWARE

Visual LinProg is an educational tool that solves LPs using animation and visualization techniques. The core of the proposed software includes the well-known class of simplex type algorithms. This tool is a web-based software and hence platform independent. It was initially designed to be used in linear programming courses for teaching simplex type algorithms. The user-student can solve his own general LPs, view the solution process step-by step, and import or export his own examples in an easy-to-read format. The solution process is covered scholastically through textual information, and the necessary steps from the pseudo code are depicted using multiple views. The graphical interpretation of a LP can be done in two modes: i) direct and ii) step-by-step. In the direct mode, the selected algorithm is executed continuously until a result (e.g., optimal, unbounded, or infeasible) is reached. In the step-by-step mode, the selected algorithm is executed systematically, and every computation of the algorithm is graphically visualized.

All the necessary computations such as, pivots, formulation, and inverse are fully animated. Furthermore, Visual LinProg was the first web-accessible tool that featured the possibility to import or export a fully customizable example. This is very important because there is not just a common predefined set of examples, which does not usually cover the students' interest. Finally, all the implemented methods (Two Phase or Big M method) are also fully detailed in order for the user to comprehend them completely.

Visual LinProg has been implemented using the Java programming language. Java applets are widely used for the development of platform-independent applications. This is very convenient in obtaining user interaction in any platform through only a browser. The only prerequisite is the installation of the Java Runtime Environment, which is crucial for the execution of any Java program. However, a vast number of educational tools have been implemented using various programming languages, e.g., Delphi in [12], Matlab in [13–14], and interactive animation development software Flash in [15], either as stand-alone executable packages [16] or as web- based applications [17].

2.1 The graphical user interface

The solution process and the presentation of the results is entirely accomplished using a Graphical User Interface (GUI) (see Figures 1 and 2) so the required mathematical symbols are shown in a common and easy-to-understand format.

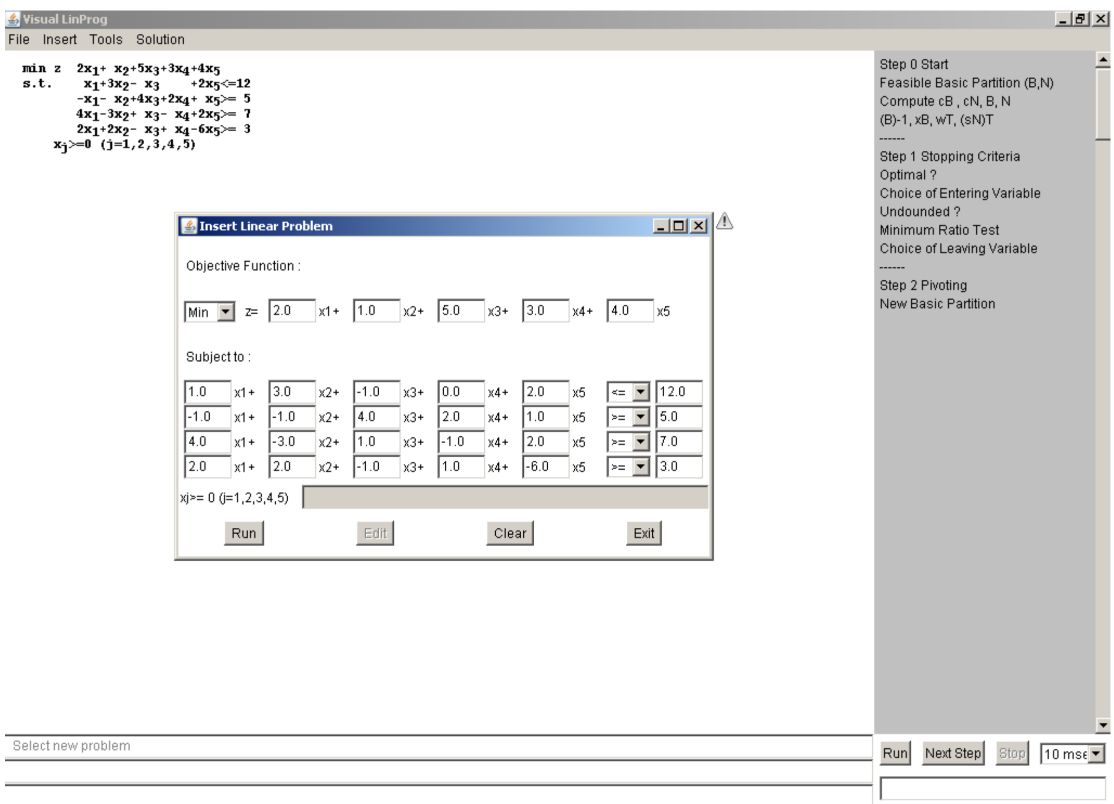

Figure 1. A screenshot depicting a user-customizable example.

In the Menu bar, the user can choose some basic operations, such as LP insertion, algorithm selection, etc. All the computations take place inside the main window, which comprises the basic workspace. All these computations, necessary for the solution of an LP, contain numerical operations, visualization, and animation.

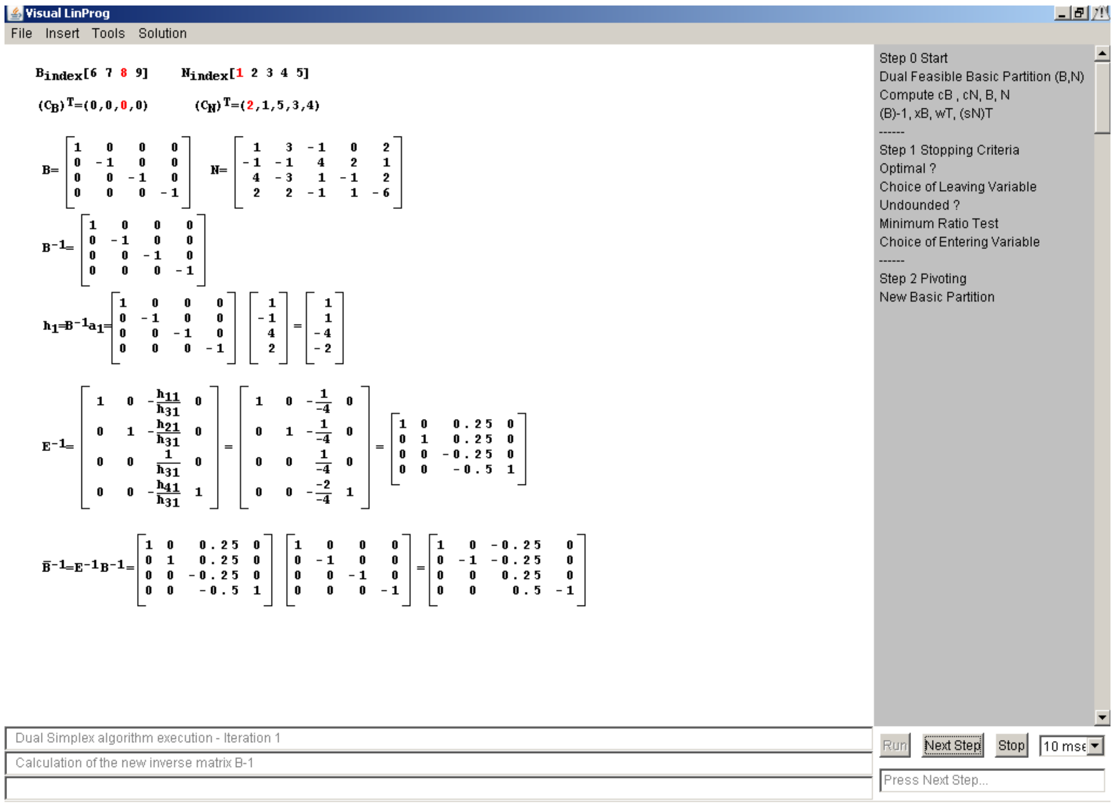

Figure 2. A second screenshot depicting the solution process.

The pseudo-code window located on the right part of the screen displays the appropriate pseudo-code, while the comments window located on the bottom part of the screen shows some explanatory messages about the solution process. In the step-by-step mode, the command line corresponding to the algorithm being executed is emphasized in red. Finally, in the control-buttons window located on the bottom right of the screen, the user can control the execution of the selected algorithms.

Researchers of algorithm visualization (AV) try to make educational software more useful and more user friendly. Thus, they work to find ways to improve the student comprehension gained from AV software. The next section presents the most essential AV software principles, as reported in the AV literature, that should be implemented in any state-of-the-art AV software. Visual LinProg, as described analytically in [7], incorporates all these principles.

2.2 Visualization software design principles

Some of the most important software principles, which influence the effectiveness of AV systems, are described in this section. To begin, the study presented by Hundhausen in [18], regarding the evaluation of various cognitive theories, shows that persons more easily remember pictures as compared to text. Therefore, the incorporation of a textual explanation of an illustration is important. Furthermore, various studies, e.g., [19–23], consider the incorporation of pseudo-code with highlighting to be important for algorithm comprehension. Pseudo-code offers a more compact way of structuring an algorithm without focusing in the precise syntax of each programming language. Moreover, students can be helped by linking to documentation. The documentation based on hypertexts gives the user control of the reading and handling of the documentation [24–27].

Additionally, the time required for the adaptation of the dynamic presentation content differs among the students, similar to the speed of comments given from a professor [22]. One of the common criticisms of the dynamic presentations is that the presentation does not give enough time for the student to think [28]. Thus, it is very helpful to provide some embedded breaks between steps. User interactivity support is also a crucial factor. Gloor [28] points out that the user should have active participation at least every 45 seconds. A simple way to achieve this is the combination of the slide transition along with the explicit implementation commands (e.g., Run, Play, etc.) that enable the student to set the desired implementation pace. However, the development of particularly dialogic animations, requiring frequent handling for the disclosure of necessary information, is rather undesirable. Gloor [25] stresses the importance of allowing the student to determine the input data himself/herself. Moreover, Gloor points out that this functionality is less important only to new users of the proposed AV application. People new to the use of an AV application find it more helpful to use a predefined set of input data rather than to import their own input data [25].

One of the ten commands of the dynamic presentation according to Gloor [28] is that there should be a description of the main steps of an algorithm. This goal can be achieved by using annotation or by providing a special control element that enumerates each logical step and also acts as a hyperlink for the transition to the current step. A similar finding is also reported in the work of Hundhausen and Douglas [8] and Khuri [25]. Both novice and advanced students might want to adjust the level of detail that is presented in an AV system. The level of detail can be modified either by hiding the corresponding details or by providing the capability to select multiple views [29].

A plethora of different matters should be taken into consideration for the sound design of the interactive environment with the user. An analytical description of the essential elements for the design and implementation of a user-interactive environment is reported in [30–31], which relate to the Human-Computer Interaction (HCI). Furthermore, smooth transitions facilitate the perception of the running operation [32]. A gradual change gives the student more time to observe the altered content. Nowadays, a plethora of researchers agree on the usefulness of smooth transitions, e.g., [21, 24, 26, 29].

Proper use of color integration can be a powerful tool in order to convey information effectively. AV applications can utilize colors in order to represent certain states of data structures, to highlight a specific activity, to connect multiple views, or to present the action's history [29]. However, research shows us that not all the color combinations are equally suitable. A wrong choice of colors could prevent, rather than strengthen, students' comprehension [30] and [25]. Moreover, focusing the student's attention is possibly the most important aspect of AV software, as already found by several educational studies [33]. Various techniques exist in order to grasp the students' attention to the main part of an algorithm as described in [22, 34].

In order for the novice students to avoid confusion, any dynamic presentation should begin with only a small data set for input. Some researchers support that, at most, seven elements should be used, or even changed, in each step [21]. This statement emanates from research in psychology, which shows that short-term memory can store at most five words or figures, six letters, seven colors, and eight digits [25]. Additionally, a restriction should exist for the number of loops. This means that the student should be able to omit the remaining part of a loop after the comprehension of its functionality. Some times, it is convenient to execute only a certain number of steps and to omit the remaining steps once the particular operation has been understood by the student.

The presentation of one interesting event per step is also an important principle. One single step in an AV system should not be overloaded with many events/processes. From the students' point of view, each step should contain at most one interesting event [21, 26]; although intermediary steps could still exist, they should not contain any other interesting event. The approach being supported by Faltin [21] is to present only interesting events. Gloor [28] considers the adjustment of the execution's speed as one more aspect of the ten commands of an algorithm's dynamic presentation. Various other reports also stress the importance of controlling the speed of the execution [25]. AV systems that are based on step-based execution and thus wait for the student's action in order to present the next step usually have fewer requirements for the control of the execution's speed. Finally, some AV systems provide the capability of saving the solution results, the entire dynamic presentation, or just a part of the presentation in cases where a complete reproduction of the dynamic presentation is not feasible. An internal, efficient saving system allows the fast retrieval of an algorithm. Nonetheless, the ability to export just part of a dynamic presentation is also a useful feature.

3. RESEARCH METHODOLOGY

3.1. Data collection and participants

The objective of this evaluation is twofold: (i) pedagogical efficiency and (ii) students' perceptions of Visual LinProg. The questionnaire was based on the evaluation from fourth-year undergraduate students during seven consecutive years (i.e., 2004-2010) at the Department of Applied Informatics in the University of Macedonia, Thessaloniki, Greece. Visual LinProg has been adopted as an educator supplement in the prerequisite course "Linear Optimization Algorithms" at the Department of Applied Informatics. Once the instructor completed the presentation of Visual LinProg and each student had used the Visual LinProg software, each participant was asked to fill out a questionnaire.

The total number of participants in our survey for all these years was 242. More specifically, there were 43, 36, 35, 44, 25, 14, and 45 participants for the years 2004, 2005, 2006, 2007, 2008, 2009 and 2010, respectively. We should note that the participants were senior students, and thus they had developed the necessary computer skills. Furthermore, no specific criteria were applied for the selection of the students (e.g., excellent grades, gender, etc.). Each student was welcomed to participate on a volunteer basis during a typical lecture in the university.

3.2. Questionnaire

The questionnaire consisted of nineteen questions and was partitioned into three parts. The first section of questions (i.e., 1-8) aims to determine the factors influencing the understanding of the Revised Simplex algorithm. More precisely, the first question group focuses on topics such as i) how difficult they consider the revised simplex algorithm in relation to other algorithms they know, ii) what kind of difficulties a student might meet while trying to understand the revised simplex algorithm, iii) how accustomed the students are to similar educational optimization software, or even AV applications, and finally iv) whether a student has any other ways to verify the correctness of a solution that he/she has computed. More specifically, the first part consisted of four yes/no questions (also known as polar questions) and four detailed questions (e.g., the students should elaborate on their answers).

The second section of questions (i.e., 9-17) seeks to find the factors influencing the effectiveness of AV. The possible answers for the questions were ranked, and students could rate the five answers using grades 1-5. Specifically, the grades were as follows: By no means = 1, Little = 2, Moderate = 3, Good = 4, and Very good = 5.

Finally, the third part consists of two detailed questions (i.e., 18-19) that try to find the student's perceptions of the effectiveness of the proposed application. These questions asks students to elaborate on i) possible drawbacks/bugs of Visual Linprog, ii) technical features that are missing from this version and should be implemented in a future version, and iii) other methods that are believed to help the understanding of similar algorithms.

However, in this subsection, our focus is on questions 9 to 17. This is due to the fact that they are more relevant to the needs of our study. These questions were specifically chosen in order to help us to determine the factors influencing the effectiveness of our AV application. These questions are as follows:

Q9. Determine the effectiveness of an AV application during the educational process.
Q10. Did the Simplex Animation software help you in the understanding of the linear optimization concepts?
Q11. Determine how user-friendly the Visual LinProg software is.
Q12. How sufficient do you believe the appearance of a problem's solution results is?
Q13. During the solution process of a problem using Visual LinProg, does the use of animated pseudo-code steps help in the comprehension of the subject being taught?
Q14. During the solution process of a problem using the Visual LinProg animation software, does the use of textual observations aid in the comprehension of the subject being taught?
Q15. How necessary, in your opinion, is the ability to import a problem for the user?
Q16. How necessary, in your opinion, is the ability to export results for the user?
Q17. Does the function of the implementation and distribution of the Visual LinProg (web-based through a common browser) assist in more effective use?

4. RESULTS, DISCUSSION and LIMITATIONS

4.1. Results

For the first section of questions, regarding the factors influencing the understanding of the revised simplex algorithm, the majority of the students (83.5%) did not know any other way to verify the correctness of a solution that he/she had computed. Furthermore, most students (68.6%) (although senior students) were not accustomed to similar educational AV applications. Additionally, we found that students, in order to learn a new algorithm, found it more helpful to i) solve many examples (69.8%) and less helpful to ii) learn the pseudo-code (19.8%) and iii) create flow charts (10.3%). Finally, half of them (57.4%) found the revised simplex algorithm more difficult than other known (e.g., sorting & searching) algorithms.

Table 1 shows the processed data (percentages) from the student replies regarding the questions of the second part. Each row and column corresponds to a numbered question and a percentage of a ranked answer, respectively. Furthermore, Table 1 is partitioned into seven subtables according to each year, while the last column shows the mean value for each question and for every year.

Table 1. Student replies from seven years: 2004-2010.

		Replies (%)					
Year	***Questions***	***1***	***2***	***3***	***4***	***5***	***Mean values***
2004: 43 participants	**Q9**	4.65	2.33	13.95	39.53	39.53	4.07
	Q10	9.30	6.98	30.23	37.21	16.28	3.44
	Q11	9.30	4.65	16.28	27.91	41.86	3.88
	Q12	0.00	2.33	13.95	51.16	32.56	4.14
	Q13	2.33	9.30	20.93	39.53	27.91	3.81
	Q14	0.00	2.33	13.95	48.84	34.88	4.16
	Q15	0.00	9.30	9.30	34.88	46.51	4.19
	Q16	0.00	4.65	23.26	30.23	41.86	4.09
	Q17	4.65	2.33	32.56	39.53	20.93	3.70
2005: 36 participants	**Q9**	0.00	0.00	0.00	19.44	80.56	4.81
	Q10	0.00	0.00	5.56	52.78	41.67	4.36
	Q11	0.00	0.00	0.00	11.11	88.89	4.89
	Q12	0.00	0.00	0.00	27.78	72.22	4.72
	Q13	0.00	11.11	2.78	27.78	58.33	4.33
	Q14	0.00	0.00	2.78	25.00	72.22	4.69
	Q15	0.00	2.78	11.11	25.00	61.11	4.44
	Q16	0.00	0.00	11.11	33.33	55.56	4.44
	Q17	0.00	2.78	25.00	22.22	50.00	4.19
2006: 35 participants	**Q9**	0.00	0.00	2.86	20.00	77.14	4.74
	Q10	0.00	2.86	2.86	42.86	51.43	4.43
	Q11	0.00	0.00	17.14	42.86	40.00	4.23
	Q12	0.00	0.00	0.00	57.14	42.86	4.43
	Q13	0.00	0.00	8.57	25.71	65.71	4.57
	Q14	0.00	0.00	5.71	34.29	60.00	4.54
	Q15	0.00	0.00	0.00	17.14	82.86	4.83
	Q16	2.86	0.00	11.43	31.43	54.29	4.34
	Q17	0.00	2.86	17.14	54.29	25.71	4.03
2007: 44 participants	**Q9**	0.00	2.27	6.82	40.91	50.00	4.39
	Q10	2.27	4.55	22.73	43.18	27.27	3.89
	Q11	2.27	2.27	18.18	25.00	52.27	4.23
	Q12	0.00	2.27	9.09	59.09	29.55	4.16
	Q13	0.00	4.55	20.45	45.45	29.55	4.00
	Q14	0.00	0.00	13.64	45.45	40.91	4.27
	Q15	0.00	2.27	6.82	27.27	63.64	4.52
	Q16	2.27	4.55	18.18	29.55	45.45	4.11
	Q17	4.55	6.82	25.00	36.36	27.27	3.75
2008: 25 participants	**Q9**	0.00	0.00	4.00	20.00	76.00	4.72
	Q10	0.00	0.00	4.00	48.00	48.00	4.44
	Q11	0.00	0.00	32.00	44.00	24.00	3.92
	Q12	0.00	0.00	4.00	52.00	44.00	4.40
	Q13	0.00	0.00	12.00	28.00	60.00	4.48
	Q14	0.00	0.00	0.00	36.00	64.00	4.64
	Q15	0.00	0.00	0.00	28.00	72.00	4.72
	Q16	0.00	0.00	16.00	28.00	56.00	4.40
	Q17	0.00	0.00	24.00	52.00	24.00	4.00
2009: 14 participants	**Q9**	0.00	0.00	0.00	64.29	35.71	4.36
	Q10	0.00	0.00	7.14	71.43	21.43	4.14
	Q11	0.00	0.00	7.14	50.00	42.86	4.36

	Q12	0.00	0.00	0.00	78.57	21.43	4.21
	Q13	0.00	0.00	7.14	50.00	42.86	4.36
	Q14	0.00	0.00	0.00	71.43	28.57	4.29
	Q15	0.00	0.00	14.29	35.71	50.00	4.36
	Q16	0.00	0.00	7.14	57.14	35.71	4.29
	Q17	0.00	0.00	14.29	71.43	14.29	4.00
2010: 45 participants	**Q9**	0.00	0.00	0.00	31.11	68.89	4.69
	Q10	0.00	0.00	8.89	31.11	60.00	4.51
	Q11	0.00	0.00	8.89	35.56	55.56	4.47
	Q12	0.00	0.00	6.67	57.78	35.56	4.29
	Q13	0.00	4.44	0.00	37.78	57.78	4.49
	Q14	0.00	0.00	0.00	48.89	51.11	4.51
	Q15	0.00	0.00	8.89	46.67	44.44	4.36
	Q16	0.00	4.44	6.67	44.44	44.44	4.29
	Q17	0.00	0.00	4.44	46.67	48.89	4.44

The estimation of the factors influencing the effectiveness of the Visual LinProg software was based on correlation analysis. A correlation represents a measure of the degree of closeness (co-relation) between two variables. Moreover, a correlation coefficient can be used as an indicator of the strength of the relationship. However, several correlation coefficients can be used, regarding the nature of data, as for example the Pearson correlation coefficient, the Kendall's tau coefficient (τ) and others. In our case, our data were categorically ranked and furthermore a frequency analysis showed a negative skew. Thus, Speaman's Rho coefficient (ρ) was a more appropriate correlation statistic, since it does not depend on the distribution of the sample data (e.g., a non-parametric measure). The Speaman's Rho coefficient, or Spearman's rank correlation coefficient, is the analogue of the Pearson product moment correlation coefficient when the data is not in interval/ ratio scale but rather in terms of ranks.

The results from the educational experiments are presented in Table 2. More precisely, we can see how question 9 correlates to the other questions through the Spearman's (ρ). We should note that the null hypothesis is $H_0 : \rho = 0$, that is, the correlation ρ between the variables is zero. Furthermore, even very weak correlations (small correlation coefficients) can be statistically significant with large sample sizes.

Table 2. Spearman's correlation coefficient ρ for the ninth (Q9) question (effectiveness of the AV program) and all other questions (factors) per annual.

		Years							
		2004	***2005***	***2006***	***2007***	***2008***	***2009***	***2010***	***Total (2004 – 2010)***
Questions	**Q10**	0.631^{**}	0.378^{*}	0.610^{**}	0.610^{**}	0.574^{**}	0.070	0.315^{*}	0.560^{**}
	Q11	0.516^{**}	0.273	0.000	0.255	-0.057	0.289	0.150	0.235^{**}
	Q12	0.507^{**}	0.009	0.171	0.315^{*}	0.060	0.337	0.392^{**}	0.362^{**}
	Q13	0.485^{**}	0.375^{*}	0.036	0.124	0.096	0.021	0.258	0.302^{**}
	Q14	0.407^{**}	0.043	0.116	0.374^{*}	0.140	0.519	0.303^{*}	0.363^{**}
	Q15	0.357^{*}	0.152	0.103	0.326^{*}	0.249	0.081	0.193	0.253^{**}
	Q16	0.351^{*}	0.182	0.175	0.290	0.182	0.126	0.033	0.222^{**}
	Q17	0.177	0.106	0.149	-0.071	0.258	0.000	0.412^{**}	0.190^{**}

In Table 2, by "**" we denote the correlation that is significant at the 0.01 level (2-tailed), while by "*" we denote the correlation that is significant at the 0.05 level (2-tailed). We should note that negative values in Table 2 indicate negative ramifications for the corresponding question to the ninth (Q9) question (effectiveness of the AV program). The same results, although in chart format, are also depicted in Figure 3.

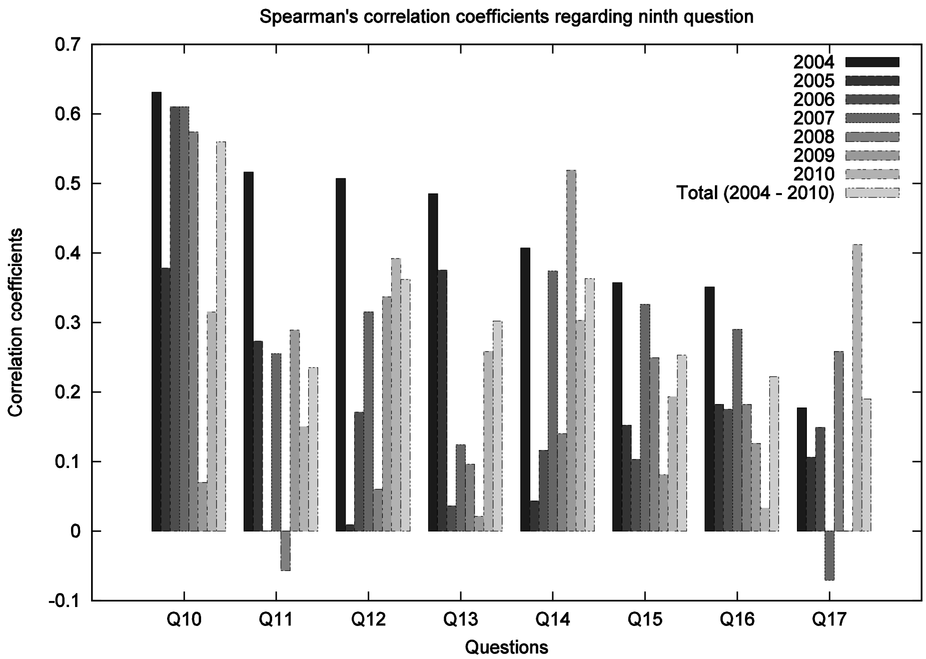

Figure 3. Spearman's correlation coefficients for the ninth (Q9) question (effectiveness of the AV program) and all other questions (factors) per annual.

The perceived usability is positively related to user satisfaction. Thus, the last column of Table 2 shows some, statistically significant, interesting results as follows:

- Students consider Visual LinProg to be useful because they are determined that it will help them to understand the revised simplex algorithm. This fact can be verified from the correlation coefficient of questions Q9 and Q10, ($\rho = 0.560^{**}$, $p < 0.001$).
- A second important factor influencing the effectiveness is the use of textual observation aids during the solution process of a problem in the Visual LinProg animation software. This fact can be verified from the correlation coefficient of Q9 and Q14, ($\rho = 0.363^{**}$, $p < 0.001$).
- The appearance of a problem's solution results also constituted a third important factor that influenced the effectiveness. The correlation coefficient of Q9 and Q12 is ($\rho = 0.362^{**}$, $p < 0.001$).
- A fourth factor influencing the effectiveness is the use of animated pseudo-code steps, during the solution process of a problem using Visual LinProg. The correlation coefficient of Q9 and Q13 is ($\rho = 0.302^{**}$, $p < 0.001$). Saraiya et al. in [35] have also stressed the importance of this factor.
- The ability to import user-customized problems is found to be a fifth factor. The correlation coefficient of Q9 and Q15 is ($\rho = 0.253^{**}$, $p < 0.001$).
- A sixth factor was found to be the user friendliness of Visual LinProg. This fact can be verified from the correlation coefficient of Q9 and Q11, which is ($\rho = 0.235^{**}$, $p < 0.001$).
- Almost of equal importance with the two previously mentioned factors is the ability to export user-customized problems. Indeed, the correlation coefficient of Q9 and Q16 is ($\rho = 0.222^{**}$, $p < 0.001$).
- Finally, the function of the implementation and distribution of Visual LinProg (web-based through a common browser) does not seem to influence the effectiveness of AV. This fact

can be verified from the correlation coefficient of Q9 and Q17, which is ($\rho = 0.190^{**}$, p = 0.003).

In the following, we present some more results on factors influencing the understanding of the revised simplex algorithm by the students/users of the Visual LinProg software. The results are presented in Table 3. More precisely, we can see how the tenth question correlates to the other questions through the Spearman's (ρ).

Table 3. Spearman's correlation coefficient ρ for the tenth (Q10) question (understanding of the revised simplex algorithm) and all other questions (factors) per annual.

		Years							
		2004	***2005***	***2006***	***2007***	***2008***	***2009***	***2010***	***Total (2004 – 2010)***
Questions	***Q9***	0.631^{**}	0.378^{*}	0.610^{**}	0.610^{**}	0.574^{**}	0.070	0.315^{*}	0.560^{**}
	Q11	0.318^{*}	0.087	0.113	0.301^{*}	0.043	0.347	0.129	0.203^{**}
	Q12	0.366^{*}	0.095	0.239	0.265	0.157	0.191	0.154	0.257^{**}
	Q13	0.505^{**}	0.582^{**}	0.456^{**}	0.392^{**}	0.467^{*}	0.569^{*}	0.233	0.496^{**}
	Q14	0.186	0.394^{*}	0.203	0.259	0.262	0.445	0.215	0.329^{**}
	Q15	0.151	0.031	0.267	0.229	0.203	-0.078	0.016	0.141^{**}
	Q16	0.300	0.206	0.190	0.191	0.184	-0.320	0.103	0.180^{**}
	Q17	0.049	0.296	0.231	-0.016	0.436^{*}	0.272	0.189	0.238^{**}

The same results, although in chart format, are also depicted in Figure 4.

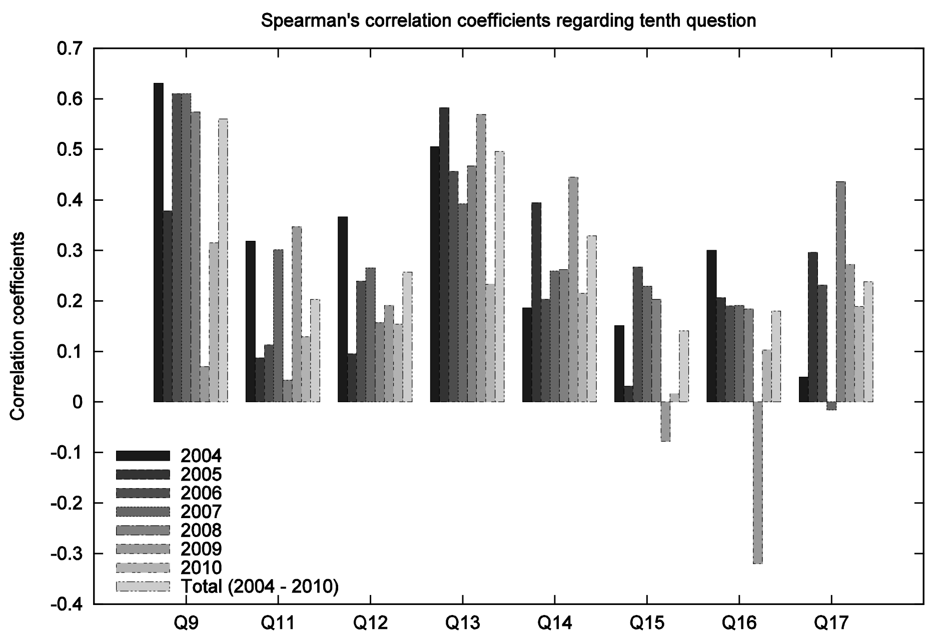

Figure 4. Spearman's correlation coefficients for the tenth (Q10) question (understanding of the revised simplex algorithm) and all other questions (factors) per annual.

The last column of Table 3 shows some, statistically significant, interesting results as follows:

- The most important factor influencing the understanding of the revised simplex algorithm for the students is the use of animated pseudo-code steps during the solution process of a problem using Visual LinProg. The correlation coefficient of Q10 and Q13 is ($\rho = 0.496^{**}$, $p < 0.001$).

- A second factor influencing the effectiveness is the use of textual observation aids during the solution process of a problem in the Visual LinProg animation software. This fact can be verified from the correlation coefficient of Q10 and Q14, which is ($\rho = 0.329^{**}$, $p < 0.001$).

Figure 5 shows the (arithmetic) mean values of the ninth question (determine the effectiveness of the AV program) and the tenth question (did Visual LinProg assist you in the understanding of the revise simplex algorithm?). It can be seen that the majority of students found Visual LinProg not only to be useful but also reported that they were helped by this tool.

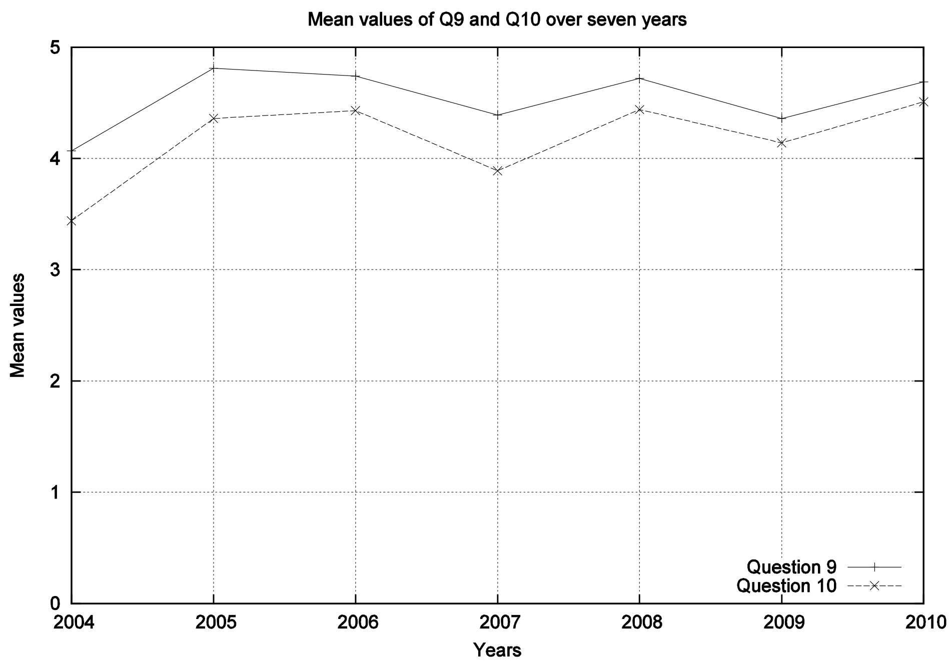

Figure 5. Mean values of questions nine and ten.

Furthermore, the weighted (arithmetic) mean values of all the questions of the second part of the questionnaire for all seven years were 4.54, 4.17, 4.28, 4.34, 4.29, 4.44, 4.49, 4.28, and 4.02, respectively, for questions 9 to 17.

4.2. Discussion

A large number of students did not participate in the lecture in which the evaluation took place, although they attended most lectures of the course “Linear Optimization Algorithms” in our university. These students were aware of the Visual LinProg application due to the description available on the web page of the course; however, we did not get their feedback. Additionally, several of those students (not participating in the evaluation) reported after the exams that they benefitted from the use of Visual LinProg. More precisely, 86 students took the exams; 47 students (54.65%) passed the exams, while the remaining 39 students (45.35%) didn’t get a passing grade. It is interesting to see that 43 out of those 47 students, who pass the exams, stated that they have used Visual LinProg during the semester.

Obviously, the students who passed the exams without having participated in the evaluation learned how to use the proposed tool independently, thus making this study a partially uncontrolled experiment. Moreover, since those students provided us with oral feedback after having used Visual LinProg for at least one semester, this study can be also considered a longitude study. These two methods (e.g., longitudinal studies of effectiveness, use of evaluation methods other than controlled experimentation) have been suggested by Stasko & Hundhausen in their study [1] as some of the most promising directions for future research in the area of AV software.

Additionally, the answers of the students regarding the third part of the questionnaire revealed some drawbacks in the current version of Visual LinProg. More specifically, these comments provide us valuable guidelines in order to develop and improve further the proposed software. Thus, the (missing) technical features that the students would like to see in the future, in frequency order, are the following:

- Efforts should be made to simplify the (rather) complicated procedure of saving.
- Provide a graphical solution of LPs.
- Enable sound.
- Supply Visual LinProg with more comments on the selection of the feasible basic partition (only for simplex-type algorithms).
- Ability to step back during the algorithm's execution.
- Solve real-world problems.
- Internationalization.
- Implement a standalone version of Visual LinProg.

Especially, the request for the solution of real-world examples cannot be considered for future implementation because real-world examples require the solution of large-scale optimization problems, and this functionality is implemented only in commercial state-of-the-art solvers and not in educational software.

Although, the majority of students did not answer the third section of questions, we strongly believe that these are very crucial questions. Should any AV software developer want to improve his/her tool, then he/she must be provided with certain feedback from the students who use it.

4.3. Limitations

It is worth mentioning that for 2006, it was not feasible to present how to import and export data files using Visual LinProg. In order to be able to present this functionality, we should modify the Java policy tool through the control panel of each PC. However, due to the fact that, especially for that year, we were provided with limited accounts in the computer lab of our university, we could not present this functionality. Therefore, the students that year did not appreciate this functionality. This observation has been reflected in the students' answers and can be seen in Figure 3.

Moreover, it can be seen that the number of participants during 2009 was the smallest for the seven years. This decrease can be partially explained due to the fact that the evaluation took place during the last lecture of the spring semester. Unfortunately, a large number of students preferred to study at home for the coming exams in June rather than to attend the last lecture.

8. CONCLUSIONS and FUTURE WORK

The scientific contribution of our empirical study is the presentation of some new results on the factors influencing the effectiveness of AV applications and the factors influencing the understanding of the revised simplex algorithm. By examining the Spearman's correlation coefficients, questions 10, 14, 12, and 13 were found to be more related to question 9. Therefore, the facts that i) students were assisted by the proposed AV application in the algorithms' comprehension (e.g., question 10); ii) textual observation aids were used during the solution process of a problem using the Visual LinProg animation software (e.g., question 14); iii) a proper appearance of a problem's solution results was used (e.g., question 12); and finally iv) Visual LinProg featured animated pseudo-code steps (e.g., question 13) constitute crucial factors influencing the effectiveness of the Visual LinProg application. Furthermore, it was found that the most important factors influencing the understanding of the proposed algorithm by the students is the use of animated pseudo-code steps and of textual observation aids during the solution process of a problem in Visual LinProg.

The acceptance of Visual LinProg by the students of our department seems to be very encouraging. A large number of students declared that they were helped greatly by using the proposed tool, and moreover, they managed to obtain passing grades in the corresponding course. However, our results constitute only one step towards this direction, and similar studies focusing on this subject would be quite welcomed. It would be very interesting to study the same factors, based on different AV software, and to check if our findings are independent of the current AV application and thus verified by other researchers. Moreover, based on our experience from the experiments presented here, we would like to bring some questions to the attention of AV researchers interested in developing their own AV software:

- Target audience of educational AV software
  If an AV application is designed as an applet, then due to the unique way that the applets can be viewed (e.g., via Internet), it is obvious that the only users will be those having Internet access. Therefore, before designing or developing the program, one should first decide what type of users will be interested in using the application. This is also known as the "target audience". Usually, due to the complexity of the algorithms that are presented, AV software is aimed at students. Therefore, AV programmers should have this type of user in mind during the development and implementation of such AV programs.
- Prerequisite knowledge
  One important parameter that should be taken into consideration is the students' potentiality for the comprehension of new knowledge. The ability to use an AV application is one way to test the easiness of the learning and utilisation of an AV system. However, due to a large divergence in the abilities of students' comprehension, an AV application is usually designed for those having the least potential for new algorithm comprehension.
- Educational value of AV applications
  The fundamental objective of AV applications is the "educational value." But what really is the meaning of that? There is absolutely no reason to use something from which one

does not learn anything. The aim of AV applications is to help students to comprehend algorithms' functionality or possibly just a part of an algorithm. This aim is accomplished through a visual representation of repeated pseudo-code commands. If the student can comprehend the basic idea of an algorithm operation by watching its corresponding visual representation, then he/she acquires the ability to apply the algorithm on his/her own. This is done because the student is in position to utilise efficiently the parallelism between the previously watched algorithms' implementation and the way that the algorithms' operation was impressed in his/her memory. Thus, by increasing the students' self-confidence, they can more easily apply algorithmic techniques in every-day problems (e.g., occurring either in their studies or in their jobs).

Finally, a possible future research direction is to determine the factors influencing the effectiveness of blended learning strategies and synchronous & asynchronous e-tools complementary to f2f traditional teaching [36].

REFERENCES

[1] J. Stasko and C. D. Hundhausen, Algorithm visualization. In: S. Fincher and M. Petre (Eds), Computer Science Education Research, Taylor & Francis, Lisse, The Netherlands, 2004, pp. 199–228.

[2] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko, A Meta-Study of Algorithm Visualization Effectiveness, J Vis Lang Comput 13(3) (2002), 259–290.

[3] J. Á. Velázquez-Iturbide, C. Pareja-Flores, and J. Urquiza-Fuentes, An approach to effortless construction of program animations, Comput Educ 50(1) (2008), 179–192.

[4] M. D. Byrne, R. Catrambone, and J. T. Stasko. Evaluating animations as student's aids in learning computer algorithms, Comput Educ 33(4) (1999), 253–278.

[5] C. D. Hundhausen, Integrating algorithm visualization technology into an undergraduate algorithms course: ethnographic studies of a social constructivist approach, Comput Educ 39(3) (2002), 237–260.

[6] C. D. Hundhausen and J. L. Brown, Designing, visualizing, and discussing algorithms within a CS 1 studio experience: An empirical study, Comput Educ 50(1) (2008), 301–326.

[7] V. Lazaridis, K. Paparrizos, N. Samaras, and A. Sifaleras, Visual LinProg: A Web-based Educational Software for Linear Programming, Comput Appl Eng Educ 15(1) (2007), 1–14.

[8] C. Hundhausen and S. Douglas, Using Visualizations to Learn Algorithms: Should Students Construct Their Own, or View an Expert's?, In: Proceedings of the 2000 IEEE International Symposium on Visual Languages (VL'00), 2000, pp. 21.

[9] S. Hansen, N. H. Narayanan, and M. Hegarty, Designing Educationally Effective Algorithm Visualizations, J Vis Lang Comput, 13(3) (2002), 291–317.

[10] C. Kann, R. W. Lindeman, and R. Heller, Integrating algorithm animation into a learning environment, Comput Educ, 28(4) (1997), 223–228.

[11] S. R. Hansen, and N. H. Narayanan. On the role of animated analogies in algorithm visualizations, In: Proceedings of the 4th International Conference of the Learning Sciences, Mahwah, NJ: Erlbaum, 2000, pp. 205-211.

[12] O. Akkoyun, Development of a visual courseware for surface mining education, to be published in Comput Appl Eng Educ (2009).

[13] J. J. C. Fernández, E. S. Olivas, J. G. Sanchis, A. J. Serrano, A. Soriano-Asensi, and J. F. G. Martínez, MATLAB-based educational software for exploratory data analysis (EDA toolkit), to be published in Comput Appl Eng Educ (2009).
[14] C. Bayilmis, Development of a web-based educational interface using MATLAB builder NE with Web Figure for digital modulation techniques, to be published in Comput Appl Eng Educ, (2010).
[15] Z. Gao and C. Wang, Constructing virtual hydraulic circuits using Flash, to be published in Com App Eng Educ, (2009).
[16] Gregoriou G., Kirytopoulos K. and Christos Kiriklidis, Project management educational software (ProMES), to be published in Comput Appl Eng Educ, (2010).
[17] Karagiannis, P., I. Markelis, K. Paparrizos, N. Samaras, A. Sifaleras, E-learning technologies: Employing Matlab Web server to facilitate the education of mathematical programming, Internat. J. Math. Ed. Sci. Tech. 37(7) (2006), 765–782.
[18] C. D. Hundhausen, The search for an empirical and theoretical foundation for algorithm visualization, Unpublished technical report, Department of Computer & Information Science, University of Oregon, Eugene, OR, 1993.
[19] L. Stern, H. Sondergaard, and L. Naish, A Strategy for Managing Content Complexity in Algorithm Animation, ACM SIGCSE Bull 31(3) (1999), 127–130.
[20] A. I. Concepcion, N. Leach, and A. Knight, Algorithma 99: An Experiment in Reusability & Component Based Software Engineering, ACM SIGCSE Bull 32(1) (2000), 162–166.
[21] N. Faltin, Aktives Lernen von Algorithmen mit interaktiven Visualisierungen. Tagungsband "Informatiktage 2000" der Gesellschaft f ̈ur Informatik, (2000), 121–137.
[22] G. Roßling and Freisleben, B., Experiences in Using Animations in Introductory Computer Science Lectures, ACM SIGCSE Bulletin 32(1) (2000), 134–138.
[23] M. Lattu, V. Meisalo, and J. Tarhio, A visualisation tool as a demonstration aid, Comput Educ 41(2) (2003), 133–148.
[24] J. M. Anderson and T. L. Naps, A Context for the Assessment of Algorithm Visualization System as Pedagogical Tools, In: Proceedings of First International Program Visualization Workshop, University of Joensuu Press, Porvoo, Finland, 2001, pp. 121–130.
[25] S. Khuri, A User-Centered Approach for Designing Algorithm Visualizations, In: J. Á. Velázquez-Iturbide and C. Pareja-Flores (Eds.), Special Issue on Visualization of Software, Joint issue of UPGRADE with Novática and Informatik/Informatique 2(2) (2001), 12–16.
[26] T. L. Naps, Incorporating Algorithm Visualization into Educational Theory: A Challenge for the Future, In: J. Á. Velázquez-Iturbide and C. Pareja-Flores (Eds.), Special Issue on Visualization of Software, Joint issue of UPGRADE with Novática and Informatik/Informatique 2(2) (2001), 17–21.
[27] R. J. Ross, Hypertextbooks for the Web, In: Proceedings of First International Program Visualization Workshop, University of Joensuu Press, Porvoo, Finland, 2001, pp. 221–233.
[28] P. A. Gloor, User Interface Issues for Algorithm Animation, In: J. T. Stasko, J. Domingue, M. H. Brown & Price B. A. (Eds.), Software Visualization - Programming as a Multimedia Experience, MIT Press, Cambridge, MA, 1997, pp. 145–152.
[29] M. Brown and J. Hershberger, Fundamental Techniques for Algorithm Animation Displays, In: J. T. Stasko, J. Domingue, M. H. Brown & Price B. A. (Eds.), Software

Visualization - Programming as a Multimedia Experience, MIT Press, Cambridge, MA, 1997, pp. 81–101.

[30] C. Faulkner, The Essence of Human-Computer Interaction, Prentice Hall, 1998.

[31] A. J. Dix, J. E. Finlay, G. D. Abowd, and R. Beale, Human Computer Interaction (3rd ed.), Prentice Hall, London, 2004.

[32] J. T. Stasko, Smooth Continuous Animation for Portraying Algorithms and Processes, In: J. T. Stasko, J. Domingue, M. H. Brown, and B. A. Price (Eds.), Software Visualization - Programming as a Multimedia Experience, MIT Press, Cambridge, MA, 1998, pp. 103–118.

[33] S. Grissom, M. McNally, and T. Naps, Algorithm visualization in CS education: comparing levels of student engagement, In: Proceedings of the 2003 ACM Symposium on Software Visualization (SoftVis ’03), San Diego, California, 2003, pp. 87–94.

[34] B. Braune and R. Wilhelm, Focusing in Algorithm Explanation, IEEE Transactions on Visualization and Computer Graphics 6(1) (2000), 1–7.

[35] P. Saraiya, C. A. Shaffer, D. S. McCrickard, and C. North, Effective features of algorithm visualizations. ACM SIGCSE Bull. 36(1) (2004), 382–386.

[36] E. Rossiou and A. Sifaleras, Blended Methods to Enhance Learning: An Empirical Study of Factors Affecting Student Participation in the use of e-Tools to Complement F2F Teaching of Algorithms, In: Proceedings of the 6th European Conference on e-Learning, Copenhagen, Denmark, 2007, pp. 519–528.