Evaluating the Effect of Program Visualization on Student Motivation

J. Ángel Velázquez-Iturbide, *Senior Member IEEE*, Isidoro Hernán-Losada and Maximiliano Paredes-Velasco

***Abstract*—An increase in student motivation is often cited as an expected effect of software visualization, but, as far as the authors are aware, no controlled experiments have yet demonstrated this. This paper therefore presents a controlled evaluation of this effect, conducted within the framework of self-determination theory. Students were tasked with removing redundant recursion in a multiple recursive algorithm. The experimental group (*N*=19) used the SRec program visualization system, while the control group (*N*=17) could use any programming tool in which they were proficient, such as BlueJ or Eclipse. The increase in motivation was measured by the difference between student scores in pre- and post-tests of their motivation. Statistically significant increases were observed for the two more determined forms of motivation, intrinsic motivation (*p*=.01) and extrinsic motivation via identified regulation (*p*=.03). Student feedback about their subjective acceptance of SRec reinforced these results. These findings give experimental support to expectations for software visualization in terms of motivation. Several paths for future work are identified, in particular, the need for more experiments under various conditions to achieve deeper understanding of the motivational effect of program visualization.**

***Index Terms*—Computer science education, educational software, experimental research, document analysis, program visualization, self-determination theory**

I. INTRODUCTION

The proverb "a picture is worth a thousand words" explains the emergence of software visualization in recent decades as a major line of research in computer science education [1]. A more accurate claim is that "a picture is *sometimes* worth a thousand words" [2]. Visualization has associated difficulties that are not evident, such as having to learn the meaning of graphical notation, and not taking this for granted [3].

A consensus in the research community is that, for educational success, the form of the learning activity in which software visualization is used is more important than is the quality of the visualizations [4]. A related factor is students' engagement with visualizations. The "engagement taxonomy" proposes several classes of student engagement: viewing, responding, changing, constructing and presenting [5]. This is probably the most commonly used evaluation framework, but additional analyses have suggested that the level of engagement is not the only factor that effects educational success [6]. The search for additional factors is ongoing and has led to several refinements or extensions of the engagement taxonomy [7]–[9]. Specific features of visualizations that are educationally effective have been identified, but even that is still an open issue [10].

It is important to note that most evaluations of algorithm visualization were focused on "effectiveness". The meta-study by Stasko, Douglas and Hundhausen [4] defines effectiveness in terms of knowledge (conceptual or procedural), and analyzes evaluations conducted as controlled experiments. In a subsequent review, also conducted by Stasko and Hundhausen [11], their analysis of effectiveness was extended to other evaluation methods (observational studies, questionnaires and surveys, ethnographic field studies, and usability studies). Only a few studies address other concerns: Ebel and Ben-Ari studied the effect of visualization on students' attention [12], and Ben-Bassat and Ben-Ari [13] studied teachers' attitudes to adopting visualization systems in their courses.

There is a surprising gap in the current literature on evaluation of software visualization. Motivation is probably the most often-cited effect on students of using visualizations. But to the authors' knowledge, the evidence is only anecdotal, with, at best, questionnaires and surveys being used to gather "preferences or opinions", that is, subjective data. A representative example of this kind of study was conducted by Stasko himself [14], who asked students to develop algorithm visualizations and rate several statements using a Likert-type scale. Students gave very high ratings to the utility of animations as a learning experience, to their usefulness in understanding algorithms, to them being characterized as fun, and to their ease of use. Obviously, students enjoyed the visualizations, but the results only have value as subjective opinions.

The situation described here is well represented in a quotation [15]: "*In course evaluations, students rate the visualizations as a fun and productive part of the course. While we have not conducted formal studies as to the ICV's effect on student learning, the positive student reaction and the evidence in educational literature on the effects of active learning approaches have convinced us that this is a productive approach.*" Note the association of formal studies with students' performance,

This paragraph of the first footnote will contain the date on which you submitted your paper for review.
This work was partially supported by research grants TIN2015-66731-C2-1-R of the Spanish Ministry of Economy and Competitiveness, S2013/ICE-2715 of the Government of the Region of Madrid, and 30VCPIGI15 of the Universidad Rey Juan Carlos.
J.Á. Velázquez-Iturbide is with the Universidad Rey Juan Carlos, 28933 Móstoles, Madrid, Spain (e-mail: angel.velazquez@urjc.es).
I. Hernán-Losadais with the Universidad Rey Juan Carlos, 28933 Móstoles, Madrid, Spain (e-mail: isidoro.hernan@urjc.es).
M. Paredes-Velasco is with the Universidad Rey Juan Carlos, 28933 Móstoles, Madrid, Spain (e-mail: maximiliano.paredes@urjc.es).
The three authors contributed equally to this work.

and the informal treatment of their motivation.

Therefore, the research question addressed in this article is whether visualization has an impact on students' motivation, stating the following hypothesis:

H. Students are more motivated in their programming tasks by using visualization tools than by using traditional tools.

The main contribution of this paper is an evaluation, in a controlled experiment based on objective data and calibrated questionnaires, of the impact of visualization on student motivation.

The experiment has some distinctive features. Firstly, the approach known as "program visualization" was adopted. Here, note the difference between two related terms [16]: program visualization and algorithm visualization. The former is concerned with visualizing actual code or data structures, while the latter is concerned with more abstract descriptions. Notice that tight linking of program visualization with code naturally leads to automation: running a program can produce, as a side effect, a visualization by rendering the program state into a graphical representation. However, the higher abstraction level of algorithm visualization usually demands the intervention of a human agent to construct the visualizations. In the evaluation, the program visualization system SRec, aimed at visualizing recursion [17], was adopted, so any user-defined recursive algorithm could be visualized using SRec.

Secondly, SRec not only supports understanding and analysis tasks, but also provides limited support to algorithm construction tasks. In particular, SRec can assist in the conversion of redundant recursive algorithms into efficient ones [18]. This process is supported by adequate graphical representations, and by the user's interaction with such representations.

This paper is structured as follows. Section II gives relevant background for the experiment. Sections III and IV present the experimental design and the results obtained, respectively. Section V presents additional evidence, gathered after the experiment, that reinforces the evaluation results. Sections VI and VII discuss the results and draw conclusions.

II. Background

This section introduces three elements of the overall evaluation presented here. First, the main features of the SRec program visualization system are summarized. Second, the use of SRec to convert redundant algorithms into efficient ones is shown. Finally, self-determination theory, a framework for analyzing motivation in educational contexts, is briefly introduced.

A. An Overview of SRec

SRec [17] is a program visualization system aimed at displaying recursive processes coded in Java. The system supports a number of representations; this study focuses on recursion trees [19], dependency graphs [20] and tables [20]. The user typically interacts with SRec by iterating the following process: 1) load a Java class; 2) select a method; 3) launch the execution of one or several test cases; and 4) interact with the visualizations generated.

SRec's user interface consists of three panels, namely the editor panel and two visualization panels. Fig. 1 shows a screenshot of the system user interface, where the editor panel had been hidden to make room for visualizations. The user launched the execution of the recursive definition of Fibonacci numbers for eight test cases, ranging from *fib*(3) to *fib*(10). The bottom visualization panel shows eight miniatures of the recursion trees generated. The sixth (in the rectangle frame), corresponding to *fib*(8), is displayed in the top panel at a larger scale.

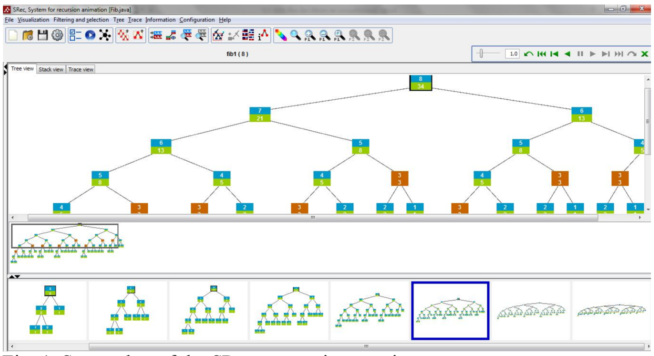

Fig. 1. Screenshot of the SRec system in a session.

Note in the top panel that the tree view was designed to cope with large trees by means of an overview+detail interface. The overview window allows the user to navigate through a large tree without becoming disoriented. The part of the tree framed in the overview window is displayed on a larger scale in a detail window, so that details can be read comfortably. Furthermore, the bottom panel can be hidden to make room for the top panel.

In general, first attempts at generating a visualization are not completely satisfactory, so the user must interact with the visualization to tailor it to requirements. SRec provides a comprehensive set of interaction facilities [21]; a detailed account of these is not given, for reasons of space, but the interactions used to obtain Figs. 2 and 3 are identified.

Finally, SRec provides several educational facilities, including that of exporting visualizations into graphical files, which provided most of the figures shown here. SRec is freely available at: http://www.lite.etsii.urjc.es/srec/ and only requires a Virtual Java Machine.

B. Support for a Programming Task: Removal of Redundant Recursion

Typically, visualization systems are intended to assist students in understanding or analyzing individual algorithms, or in comparing the performance of different algorithms that solve the same problem. The authors are not aware of any visualization system that assists students in construction tasks. SRec is therefore an innovative visualization tool in that it assists programmers in a specific case of algorithm construction [18], that is, the systematic conversion of redundant, multiple recursive algorithms into equivalent, efficient algorithms.

Redundant, multiple recursive algorithms have prohibitive algorithmic complexity. Unless these algorithms are applied to input data of very small size, redundancy must be removed. This can be found, for instance, in the recursive equations de-

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TE.2017.2648781

signed in a first stage of dynamic programming solutions. The recursion removal methodology is not described here in detail, for reasons of space, but the interested reader can find either informal descriptions in algorithm textbooks [22], [23], formal descriptions in technical publications [20], or a more detailed account of the support provided by SRec in [18].

The methodology can be briefly illustrated with the following algorithm, implemented as Java method *f*. This method was used in the evaluation described in Section III:

```
public static int f (int y, int x) {
   if (y==0)
      return x;
   else if (x==0)
      return y;
   else
      return f(y,x-1) + f(y-1,x) + f(y-1,x-1);
}
```

If this recursive method is applied to y=2, x=4, the recursion tree of Fig. 2 is obtained. The figure only shows the tree shape, but SRec's several interaction facilities allow observation and analysis of the details of very large recursion trees. The user can perform various operations, most notably filtering (e.g., hiding parameters or methods), zooming, and exploring (by scrolling or by using the overview+detail interface shown in Fig. 1).

Although Fig. 2 is unreadable at the scale shown, it nevertheless allows the redundancy of method *f* to be detected. Redundancy can be checked by searching and highlighting equal calls. For instance, if occurrences of call *f*(1,3) are searched, three calls are highlighted as a result (see Fig. 2). SRec also allows a user to query the number of nodes in the recursion tree. The tree has 61 nodes, but the number of different recursive calls is (2+1)·(4+1)=15. This disparity is due to repeated (i.e., redundant) invocation of many recursive calls.

The next step is to analyze the recursion pattern. The recursion tree is transformed into a dependency graph (i.e., an acyclic directed graph) by joining equal nodes while preserving arcs. The dependency graph obtained for *f*(2,4) is similar to that shown in Fig. 3. In this case, the graph generated is regular, but for other algorithms graph nodes must be spatially rearranged until a meaningful pattern is identified.

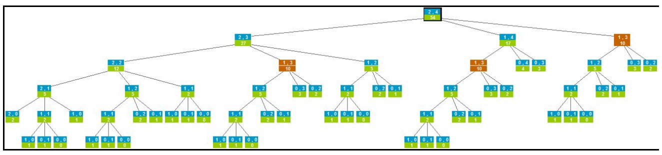

Fig. 2. Recursion tree for *f*(2,4).

The next step is to design a table that can store at least as many subproblem values as the dependency graph. Fig. 3 shows that subproblem values can be stored in a bi-dimensional array of $(x+1)\cdot(y+1)$ cells, where x and y are the argument values in the initial invocation of *f*.

Given the table, the user can decide their approach to implementing an equivalent, non-redundant algorithm, either tabulation [20] or memoization [24]. Any alternative algorithm must return the same result as the original algorithm, assisted by the table, but without producing redundant computations. Although SRec does not support this final coding phase, it is usually trivial from the table design. In particular, dynamic programming algorithms are implemented using tabulation.

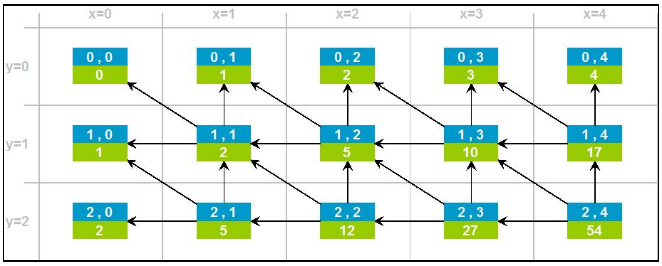

Fig. 3. Table generated from the recursion tree in Fig. 2.

C. *Self-Determination Theory*

Self-determination theory is a framework frequently used to analyze motivation in educational contexts [25]. Self-determination is based on intrinsic motivation (the manifestation of the human tendency toward learning and creativity) and on self-regulation (the way in which how people assume social values and extrinsic contingencies that are progressively transformed into personal values and self-motivation). The theory structures motivation into three different dimensions, from the highest to the lowest self-determination:

1) Intrinsic motivation. In this dimension the subject does something because it is inherently interesting or enjoyable to him/her. Intrinsic motivation is an important phenomenon for successful learning [26].
2) Extrinsic motivation. This refers to the implication of external aspects in the subject. Two sublevels can be differentiated:
   – Extrinsic motivation via identified regulation. This dimension of motivation occurs when the behavior is considered important for the subject's goals and values.
   – Extrinsic motivation via external regulation. This has less self-determination than identified regulation; here, doing something leads to a separable outcome (to obtain a reward or to avoid a punishment).
3) Amotivation. This occurs when individuals perceive no relationship between the behavior and its consequences, and behavior has no intrinsic or extrinsic motivators.

Note that, considering the two sublevels of extrinsic motivation from a practical point of view, in practice four dimensions can be differentiated.

The SIMS scale was developed to assess motivation according to self-determination theory [27]. A Spanish version of SIMS, the EMSI scale [28], was used for this study; it has 14 items grouped into the four dimensions of motivation in self-determination theory. Both scales were validated in educational contexts.

One important feature of the EMSI questionnaire is that global motivation can be computed in terms of its four constituent dimensions according to the following formula [29]:

$M = 2\cdot IM + EMI - EME - 2\cdot AM$

where M = "motivation", IM = "implicit motivation", EMI =

"extrinsic motivation via identified regulation", EME = "extrinsic motivation via external regulation", and AM = "amotivation".

III. Experimental Design

This section presents the experimental design of the controlled evaluation, giving the hypothesis, and describing the variables, the measurement instrument, the participants and the protocol followed in the evaluation.

A. Detailed Hypotheses

The previous section introduced self-determination theory and the four dimensions of motivation it differentiates. The hypothesis given in Section I can now be refined, as motivation can be measured either globally or disaggregated into dimensions. By considering the sign of each dimension contribution to global motivation (see the formula in Section II.C), four additional hypotheses can be derived, resulting in a set of five hypotheses:

H. Students are more motivated in their programming tasks by using visualization tools than by using traditional tools.

H-IM. Students are more intrinsically motivated in their programming tasks by using visualization tools than by using traditional tools.

H-EMI. Students are more extrinsically motivated via identified regulation in their programming tasks by using visualization tools than by using traditional tools.

H-EME. Students are more extrinsically motivated via external regulation in their programming tasks by using visualization tools than by using traditional tools.

H-AM. Students are more amotivated in programming tasks by using visualization tools than by using traditional tools.

B. Variables and Measurement Instrument

Obviously, the independent variable is the tool adopted to solve the assignment, either the SRec visualization tool or any conventional programming tool. The course is based on Java, with students being proficient in the use of several IDEs, at least in BlueJ and Eclipse.

The dependent variable is the increase in students' motivation. As introduced in Section II.C, the EMSI questionnaire was used to measure students' four dimensions of motivation. Students were asked to complete an EMSI questionnaire before and after the experiment. Consequently, the increase of students' motivation was defined as the motivation measured in the post-test minus the motivation measured in the pre-test. Each hypothesis is measured by subtracting the corresponding motivation scores in the pre- and the post-test.

C. Population and Educational Context

The evaluation was conducted in the academic year 2015/16 in a fourth-year elective course on advanced algorithms. The course was offered at the authors' university to computer science majors. Forty-seven students were enrolled in the course.

The course is structured around algorithm design techniques, such as branch-and-bound, dynamic programming, and approximate algorithms. The evaluation was conducted in the context of the dynamic programming technique. Students have severe difficulties in grasping dynamic programming [30]. To make it easier for them to assimilate, the technique is introduced in two steps, to separate challenging items [31]:

1) Methodology for removal of redundant recursion. The methodology outlined in Section II.B is presented in detail and exercised, adapted to derive either tabulated or memoized algorithms. The algorithms used to illustrate the methodology are numeric, such as the recursive declarations of Fibonacci or combinatorial numbers. Typically, these algorithms have regular dependency graphs [20].
2) The dynamic programming technique. Emphasis is given to the design of recursive equations. Recursion removal is accomplished exclusively by tabulation. Optimization problems are solved, for instance the 0/1 knapsack problem or the multistage graph problem. These are often intricate algorithms, with irregular dependency graphs [20].

Students had to solve one assignment per syllabus chapter, so had to solve two assignments for these two chapters. The effect of SRec on students' motivation was evaluated in the assignment on recursion removal.

D. Protocol

The evaluation was conducted in a laboratory session. The goal of these sessions was to allow students to start solving the corresponding assignment, and give them a chance to ask the instructor questions. Attendance is voluntary. Students are encouraged to get as far as possible with the assignment during the lab session, but they have one week to deliver an assignment report through the virtual campus.

For the evaluation, students were given an assignment statement containing the redundant algorithm included in Section II.B, and were asked to convert it into two efficient algorithms (a tabulated algorithm and a memoized one). Students worked individually.

Students were randomly divided into two groups, whose homogeneity of knowledge was checked to avoid motivation bias due to different knowledge levels. This was done by ensuring that grades obtained in previous assignments were similar in each group (means compared using the Student's t-test, with p-value = .17). The experimental group used SRec to solve the assignment; the control group could use any Java programming tools they had mastered (typically, BlueJ or Eclipse).

Each group worked in a different computer lab. An instructor was present in each lab; they periodically swapped labs to avoid any influence on students. Both instructors are coauthors here, but only one was involved in the development of SRec.

The two-hour session had the following schedule (with approximate timings of each phase):

1) The goal of the assignment was described, explaining that the goal of the evaluation was to enhance instruction and that participation was voluntary (3 minutes).
2) Students completed a motivation pre-test (5 minutes). The pre-test was an instantiation of the EMSI questionnaire, with the question "Why do you think that you need to solve this assignment on redundant multiple recursion?"

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TE.2017.2648781

3) All students downloaded the assignment statement available at the virtual campus (3 minutes). Students in the experimental group also downloaded SRec and installed it on their computers (3 additional minutes).
4) *For the experimental group only*, the instructor gave a quick demo of SRec (5 minutes), including: installation, launching animations, generation of visualizations (recursion trees, dependency graphs, and tables), and exporting visualizations into graphical files.
5) Students worked on the assignment (about 100 minutes).
6) Students filled in a motivation post-test (5 minutes), based on the question "Considering what you did in the session, why do you think that you need to solve this assignment on redundant multiple recursion?" For the experimental group, the question had the additional phrase "using SRec".
7) *For the control group only*, the instructor gave the same quick demo already given to the experimental group (5 minutes), and SRec was made available to them in the virtual campus.

Note that the protocol was designed to make the evaluation possible, but also to give both groups of students the same materials and opportunities to solve the assignment by the given deadline.

Although both groups had the same number of students, not all of them attended the lab. In the experimental group 19 students attended, and in the control group 17 students attended.

IV. Results

The pre-test was completed by 17 students in the control group. One student gave the same mark to all the questions in the post-test; he was felt to have answered hastily and contradictorily, so his test was discarded. Consequently, 16 valid questionnaires were left in the post-test for the control group. In the experimental group, 19 pre-test questionnaires and 15 post-test questionnaires were gathered.

Table I shows the results of global motivation obtained for the control and the experimental groups, respectively, in the pre- and post-test. Tables II and III show similar results obtained for all the dimensions of motivation.

Table I. Descriptive Statistics of global motivation for both Groups

	Control group		Experimental group	
	Pre-test (N=17)	Post-test (N=16)	Pre-test (N=19)	Post-test (N=15)
Mean	4.76	4.30	2.81	5.49
Median	5.00	3.92	3.08	6.57
Variance	26.63	24.52	10.87	27.27
Std. deviation	5.16	4.95	5.22	3.30

Table II. Descriptive Statistics of the four dimensions for the Control Group

	Pre-test (N=17)				Post-test (N=16)			
Dimensions	IM	EMI	EME	AM	IM	EMI	EME	AM
Mean	4.75	5.65	5.06	2.66	4.59	5.13	4.73	2.64
Median	5.00	5.67	5.00	2.25	4.75	5.00	5.00	2.25
Variance	1.31	0.84	2.35	1.54	0.74	1.51	1.57	1.67
Std. deviation	1.15	0.92	1.53	1.24	0.86	1.23	1.25	1.29

Table III. Descriptive Statistics of the four dimensions for the Experimental Group

	Pre-test (N=19)				Post-test (N=15)			
Dimensions	IM	EMI	EME	AM	IM	EMI	EME	AM
Mean	4.05	4.51	4.28	2.76	4.95	5.19	4.67	2.47
Median	4.00	4.59	5.33	2.75	5.00	5.22	4.67	2.25
Variance	0.59	0.61	0.86	1.17	1.22	0.89	3.57	1.28
Std. deviation	0.77	0.78	0.92	1.08	1.10	0.94	1.89	1.13

Table I shows the results of global motivation for both groups. There is an increase in the mean for the experimental group, but hardly any variation for the control group. Table II, for the control group, shows a decrease in the means of all the dimensions of motivation except amotivation between the pre- and the post-test. However, Table III shows that the experimental group exhibits an increase in the three first kinds of motivation and a decrease in amotivation.

Normality was tested to check whether mean variations were statistically significant, using a confidence interval of 95%, comparing the means obtained in the pre- and the post-test. The normality of these samples was analyzed using the Shapiro-Wilk test; the results are shown in Table IV.

Table IV. Shapiro-Wilk's Normality Tests

Group	M	IM	EMI	EME	AM
Control	.59	.31	.01	.07	.03
Experimental	.61	.93	.28	.58	.10

The p-values obtained guarantee normality in both groups for data gathered in pre- and post-tests, except in the control group for extrinsic motivation via identified regulation (p=.01) and amotivation (p=.03). Therefore, non-parametric tests [32] (Wilcoxon test) were used for these two sets of data, whereas parametric tests (Student's t-test) were applied to the remaining sets of data. The results obtained for mean contrast are shown in Table V.

Table V. Mean Contrast between pre- and post-test (p-value)

Group	M	IM	EMI	EME	AM
Control	.79	.66	.22[a]	.51	.97[a]
Experimental	.09	.01	.03	.47	.44

[a] Non-parametric test of Wilcoxon

The increase in global motivation is not statistically significant (p=.09) but is close. It is therefore convenient to make a separate analysis for each motivation dimension. A p-value greater than .05 was obtained in all disaggregated cases, except for the experimental group in the dimensions of intrinsic motivation (p=.01) and extrinsic motivation via identified regulation (p=.03). Consequently, statistical differences in the two dimensions with the highest degrees of self-determination are significant.

V. Perception of SRec

This section presents additional perception results for SRec usage. This data was obtained outside the controlled experiment at a later date, but is included because they provide additional support for the results of the controlled experiment, providing a triangulation of results.

Forty-five final assignment reports were submitted one week after the lab session. For all the assignments, the work had been carried out with SRec. Note that this number is greater than the number of students who attended the laboratory (36), because lab attendance was voluntary.

The report outline included a final section in which students were encouraged to draw their personal conclusions, and comment on any aspect of interest to them, such as any difficulties they faced in solving the assignment. The authors were surprised to notice the very large percentage of students who included comments on SRec (84.5%, see Table VI).

Table VI. Student Comments on SRec (N=45)

Comments on SRec	# students	% students
No comments	2	4.4%
Comments, not on SRec	5	11.1%
Comments about SRec and other issues	30	66.7%
Comments only about SRec	8	17.8%

The comments were analyzed qualitatively. The content analysis was conducted as an informal combination of two methods: conventional analysis [33] and theoretical reading [34]. In the former, there are no previously existing categories, as advocated by grounded theory [35]. In the latter, current knowledge is used to scaffold the identification of categories; in particular, concepts from user interaction, such as ease of use or user's effort, were considered.

Comments were decomposed into "simple comments" to facilitate the coding phase, because each comment often comprised several opinions. As a consequence, the 38 contributing students supplied 47 simple comments.

Categories emerged, from regularities and similarities between different simple comments, and these were refined in several rounds. The resulting categories are shown in Table VII. "Acceptance" denotes that students liked SRec.

Table VII. Categories of Comments written about SRec (N=47)

Categories of comments	# comments	% students
Usefulness	35	74.5%
Ease of use	6	12.8%
Acceptance	3	6.4%
Improvement suggestions	3	6.4%

Students' comments on SRec's usefulness were further analyzed. These comments were decomposed into "simple forms of usefulness", resulting in 50 simple comments. Four categories of usefulness were identified, as shown in Table VIII.

Table VIII. Categories of Usefulness in SRec (N=50)

Categories of usefulness	# comments	% students
Facilitate understanding	22	44%
Less effort	20	40%
Safety or accuracy	7	14%
Generic comment	1	2%

Notice that about half the comments claimed that SRec made understanding the various parts of the task easier, and the other half said that SRec made it possible to accomplish the assignment with less effort or faster than otherwise. These opinions were held by students regardless of which group they were in during the evaluation session.

Two eulogistic but representative comments, from two students who were in different groups during the evaluation session, follow:

> "As a conclusion, this tool is, in my opinion, the most useful of those we have used in the course" (Experimental group, student 04.)
>
> "On the other hand, I liked the SRec tool because I had never before seen a program that drew specific trees so fast for me, that is very easy to use and that will assist me in the future in other assignments and courses" (Control group, student 02.)

VI. Discussion

The controlled evaluation revealed a tendency towards increased motivation for the experimental group (p=.09) and hardly any variation for the control group. Therefore, hypothesis H was not confirmed. Actually, the increase in motivation for the experimental group was due to the two dimensions with the highest degree of self-determination. Hypotheses H-IM and H-EMI were thus confirmed (p=.01 and p=.03, respectively), but this was not the case for hypotheses H-EME and H-AM. As a reminder, in the dimension of intrinsic motivation, the subject does something because it is inherently interesting or enjoyable to him/her. Furthermore, the dimension of extrinsic motivation via identified regulation occurs when the behavior is considered important for the subject's goals and values. Given these results, it may be claimed that program visualization is a motivating educational mechanism (at least, in the specific way used for the evaluation).

The students' reports provide additional evidence of these positive results. Students were not asked to comment on any specific aspect of the assignment, but most of them praised SRec, noting its usefulness either in understanding the assignment solution or in completing the assignment with less effort or in a shorter time.

The experimental results verify (and are consistent with) the motivational expectations of educational software visualization. The reader may wonder whether the results are transferable to other visualization systems or educational contexts. That visualization is motivating *per se*, independent of the features of the visualization system or the educational use of visualization, cannot be claimed from the experimental results. Identifying the circumstances, if any, under which software visualization is motivating remains an open issue.

Several features of SRec were especially relevant in the controlled experiment and may have contributed to the positive results. Firstly, SRec partially supports an algorithm construction process (by means of conversions between several graphical representations). This is in contrast to alternative programming tools that do not support this construction methodology. Such a constructive use of visualization is uncommon and can be very appealing for programmers.

A second relevant factor could be the high interactivity of SRec. Such interactivity constitutes a powerful means of handling any situation, in particular very large trees in the task addressed in the evaluation. Debuggers present in conventional programming tools are also highly interactive, but their text-oriented format does not fit well with the task addressed in the experiment.

It would be important in the future to enquire further into the likely effect of these factors on students' motivation.

VII. Conclusion

The most notable result of the controlled experiment was a statistically significant increase in the two dimensions of motivation with the highest degree of self-determination, that is, intrinsic motivation and extrinsic motivation via identified regulation. This is especially noteworthy as it is the first formal evaluation of motivation due to visualization documented in the literature.

In addition, students' feedback in their final reports provided informal evidence and reinforced those positive results. Most students freely praised SRec and remarked on its usefulness either in understanding of the assignment solution or accomplishing the assignment with less effort or in shorter time.

The study has two limitations. Firstly, the research was conducted with a small number of participants. Secondly, the research was conducted under specific conditions of tool and task. Consequently, results cannot be straightforwardly transferred to other contexts where software visualization is used.

In the near future, several challenges can be addressed. First, in the mid term, an evaluation of students' motivation using SRec is being planned. Second, SRec is being extended to semi-automatically generate Java code from tables. Finally, it would be very interesting to replicate this study under different conditions. This evaluation was conducted for a program construction task, but other conditions can also be established, such as understanding or analyzing recursive algorithms.

References

[1] S. Fincher and M. Petre, "The field and the endeavor," *Computer Science Education Research*, S. Fincher and M. Petre, eds., London: Routledge, pp. 1-81, 2004.

[2] J. Larkin and H. Simon, "Why a diagram is (sometimes) worth 10000 words," *Cognitive Science*, vol. 11, no. 1, pp. 65-99, 1987.

[3] M. Petre, "Why looking isn't always seeing: Readership skills and graphical programing," *Communications of the ACM*, vol. 38, no. 6, pp. 33-44, Jun. 1995.

[4] C.D. Hundhausen, S.A. Douglas and J.T. Stasko, "A metastudy of algorithm visualization effectiveness," *Journal of Visual Languages and Computing*, vol. 13, no. 3, pp. 259–290, 2002.

[5] T. Naps, G. Roessling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger and J.Á. Velázquez-Iturbide, "Exploring the role of visualization and engagement in computer science education," *SIGCSE Bulletin*, vol. 35, no. 2, pp. 131-152, Jun. 2003.

[6] J. Urquiza-Fuentes and J.Á. Velázquez-Iturbide, "A survey of successful evaluations of program visualization and algorithm animation systems," *ACM Trans. Computing Education*, vol. 9, no. 2, article 9, Jun. 2009.

[7] T. Lauer, "Reevaluating and refining the engagement taxonomy," *Proc. 13th Annual Conf. Innovation and Technology in Computer Science Education (ITiCSE 2008)*, p. 355, 2008.

[8] N. Myller, R. Bednarik, E. Sutinnen and M. Ben-Ari, "Extending the engagement taxonomy: Software visualization and collaborative learning," *ACM Trans. Computing Education*, vol. 9, no. 1, article 7, Mar. 2009.

[9] J. Sorva, V. Karavirta and L. Malmi, "A review of generic program visualization systems for introductory programming education," *ACM Trans. Computing Education*, vol. 13, no. 4, article 15, Nov. 2013.

[10] P. Saraiya, C.A. Shaffer, D.S. McCrickard and C. North, "Effective features of algorithm visualizations," *Proc. 35th SIGCSE Technical Symp. Computer Science Education (SIGCSE 2004)*, pp. 382-386, 2004.

[11] J.T. Stasko, and C.D. Hundhausen, "Algorithm visualization," *Computer Science Education Research*, S. Fincher and M. Petre, eds. London: RoutledgeFalmer, pp. 199-228, 2004.

[12] G. Ebel and M. Ben-Ari, "Affective effects of program visualization," *Proc. 2nd International Computing Education Research Workshop (ICER 2006)*, pp. 1-5, 2006.

[13] R. Ben-Bassat Levy, and M. Ben-Ari, "We work so hard and they don't use it: Acceptance of software tools by teachers," *Proc. 12th Annual SIGCSE Conf. Innovation and Technology in Computer Science Education (ITiCSE 2007)*, pp. 246–250, 2007.

[14] J.T. Stasko, "Using student-built animations as learning aids," *Proc. 28th SIGCSE Technical Symp. Computer Science Education (SIGCSE 1997)*, pp. 25-29, 1997.

[15] D. Schweitzer and W. Brown, "Interactive visualization for the active learning classroom," *Proc. 38th SIGCSE Technical Symp. Computer Science Education (SIGCSE 2007)*, pp. 208-212, 2007.

[16] B. Price, R. Baecker and I. Small, "An introduction to software visualization," *Software Visualization*, J. Stasko, J. Domingue, M.H. Brown and B.A. Price, eds, Cambridge, MA: MIT Press, pp. 3-27, 1998.

[17] J.Á. Velázquez-Iturbide, A. Pérez-Carrasco and J. Urquiza-Fuentes, "SRec: An animation system of recursion for algorithm courses," *Proc. 13rd Annual Conf. Innovation and Technology in Computer Science Education (ITiCSE 2008)*, pp. 225-229, 2008.

[18] J.Á. Velázquez-Iturbide and A. Pérez-Carrasco, "Systematic development of dynamic programming algorithms assisted by interactive visualization", *Proc. 21st Annual Conf. Innovation and Technology in Computer Science Education (ITiCSE 2016)*, pp. 71-76, 2016.

[19] S.M. Haynes, "Explaining recursion to the unsophisticated," *SIGCSE Bulletin*, vol. 27, no. 3, pp. 3-6 and 14, Sep. 1995.

[20] R.S. Bird, "Tabulation techniques for recursive programs," *ACM Computing Surveys*, vol. 12, no. 4, pp. 403-417, Dec. 1980.

[21] J.Á. Velázquez-Iturbide and A. Pérez-Carrasco, "InfoVis interaction techniques in animation of recursive programs," *Algorithms*, vol. 3, no. 1, pp. 76-91, Mar. 2010.

[22] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein. *Introduction to Algorithms*, 3rd ed., Cambridge, MA: The MIT Press, chap. 15, 2009.

[23] S. Sahni *Data Structures, Algorithms and Applications in Java*, Summit, NJ: Silicon Press, chap. 20, 2005.

[24] G. Brassard and P. Bratley, *Fundamentals of Algorithmics*. Englewood Cliffs, NJ: Prentice-Hall, chap. 8, 1996.

[25] E.L. Deci, R.J. Vallerand, L.G., Pelletier and R.M. Ryan, "Motivation and education: The self-determination theory perspective," *Educational Psychologist*, vol. 26, nos. 3 & 4, pp. 325–346, 1991.

[26] R.M. Ryan and E.L. Deci, "Intrinsic and extrinsic motivations: Classic definitions

and new directions," *Contemporary Educational Psychology*, vol. 25, pp. 54–67, 2000.

[27] F. Guay, R.J. Vallerand and C. Blanchard, "On the assessment of situational intrinsic and extrinsic motivation: The Situational Motivation Scale (SIMS)," *Motivation and Emotion*, vol., 24, no. 3, pp. 175–213, 2000.

[28] J. Martín-Albo, J.L. Núñez and J.G. Navarro, "Validation of the Spanish version of the Situational Motivation Scale (EMSI) in the educational context," *The Spanish Journal of Psychology*, vol. 12, no. 2, pp. 799–807, 2009.

[29] C. Lonsdale, C.M. Sabiston, T.D. Raedeke, A.S. Ha and R.K. Sum, "Self-determined motivation and students' physical activity during structured physical education lessons and free choice periods", *Preventive Medicine*, vol. 48, pp. 69–73, 2009.

[30] H. Danielsiek, W. Paul and J. Vahrenhold, "Detecting and understanding students' misconceptions related to algorithms and data structures," *Proc. 43rd Technical Symp. Computer Science Education (SIGCSE 2012)*, pp. 21-26, 2012.

[31] F. Marton and S. Booth. *Learning and Awareness.* New York, NY: Routledge, 1997.

[32] L. Cohen, L. Manion and K. Morrison. *Research methods in education*, 7th ed, Routledge, chap. 36, 2011.

[33] H.-F. Hsieh and S.E. Shannon, "Three approaches to qualitative content analysis," Qualitative Health Research, vol. 15, no. 9, pp. 1,277-1,288, Nov. 2005.

[34] S. Kvale and S. Brinkmann, *Interviews; Learning the Craft of Qualitative Research Interviewing*, Thousand Oaks, California: SAGE, 2nd ed., chap. 14, 2009.

[35] B. Glaser and A. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Chicago: Aldine, 1967.

J. Ángel Velázquez-Iturbide received the Bachelor's degree in computer science and the Ph.D. degree in computer science from the Universidad Politécnica de Madrid, Spain, in 1985 and 1990, respectively. He is currently with the Universidad Rey Juan Carlos as a professor, where he is the leader of the Laboratory of Information Technologies in Education (LITE). His research areas include programming education and software visualization. Prof. Velázquez is a senior member of the IEEE Computer and Education Societies, and a senior member of ACM. He is the President of the Spanish Association for the Advancement of Computers in Education (ADIE).

Isidoro Hernán-Losada received the Ph.D. in computer science and mathematical modeling from the Universidad Rey Juan Carlos, Spain, in 2012. He is currently an assistant professor at the Universidad Rey Juan Carlos. His research interests are software for programming education, and human-computer interaction.

Maximiliano Paredes-Velasco received the Ph.D. in computer science from the Universidad Castilla – La Mancha, Spain, in 2006. He is currently an assistant professor at the Universidad Rey Juan Carlos. His research interests include different computer supported learning fields (collaborative learning, active learning and mobile learning) and human-computer interaction.