A Top-Down Approach to Algorithm Animatio

Margaret-Anne D. Storey [*†]
F. David Fracchia [*]
Sheelagh Carpendale [*‡]

School of Computing Science
Simon Fraser University
Burnaby, B.C. V5A 1S6 CANADA

Technical Report CMPT 94-05
September 1994

ABSTRACT

Algorithm animations have proven to be successful for the presentation and explorati
gorithms. However, when viewing an animation of a complex algorithm, the amount
displayed can be overwhelming. In an effort to increase the level of comprehension, t
introduces *expandable buttons* as a new interface approach for animating algorithms. Ex
buttons allow for top-down explorations of algorithms by providing interactive control
amount of detail displayed and visually integrating the algorithm with the animation. I
the algorithm becomes the user interface.

Keywords: expandable buttons, top-down approach, algorithm animation, user interfa

[*] Graphics and Multimedia Lab.
[†] Rigi Lab, University of Victoria.
[‡] Algorithms Lab.

1 Introduction

Traditionally, algorithms are presented using increasing levels of detail and complexity. in the classroom or textbook, the algorithm name and purpose are first introduced, fol an overview of the main steps in the algorithm. The presenter then guides the student the complexities of the algorithm by providing further details about each of these ste result is a top-down presentation; an approach which has historically worked well in th and presentation of large software systems.

A drawback of this traditional approach is that the student is constrained to follow the c pace of the presentation chosen by the instructor or author. The approach is static and address the diversity of each individual's skill and learning.

A more dynamic approach is the use of animations which allow for the exploration of alg According to Stasko [12], algorithm animation "is the process of abstracting a progran operations, and semantics, and creating dynamic graphical views of those abstraction animation, or visualization, should allow for multiple levels of abstraction and be custon

A number of successful animation environments exist, but are often inadequate, particula animating complex algorithms. One problem is that the animations often present too mu for the average user to digest. Tools may exist for altering the granularity of the detail p but more often than not, the user needs some prior knowledge of the algorithm or guida this.

Often images are used when text would have been a more appropriate choice [9]. Th informative algorithm animations, is for text to complement pictures. This is importa user is to associate steps in the algorithm with animation events. Some systems prese of program code in conjunction with the animation [4]. However, making a direct as between these views is often difficult, especially for large, complex examples [3].

This paper introduces *expandable buttons* as a new interface approach for animating alg This method integrates the traditional top-down approach with the dynamic capabilit teractive computer animation. Steps and substeps of the algorithm are visually linked resulting animation by representing each step in terms of a user interface widget. This a the selection of which (sub)steps to animate and permits the user to interact with the a through the algorithm itself. In essence, the algorithm has become the user interface.

Section 2 outlines the necessary features of an effective algorithm animation with refe existing systems. The top-down approach is described in Section 3 and illustrated thr animation of a complex algorithm in Section 4. Finally, Section 5 discusses the advant limitations of this approach, and outlines future work.

2 Requirements for an Effective Algorithm Anima

The development of the top-down approach relies on the identification of the necessary ments for an effective algorithm animation; specifically those targeted towards a goal or in example, the requirements for teaching an algorithm will not be the same as those for in a student how to write algorithms.

Algorithms are animated or visualized for a variety of reasons. One is to teach student write algorithms; the Novice's Algorithm Teacher (NAT) [5] system was designed to assist enrolled in an introductory computer science course. Another is to aid in the design and of algorithms; the Balsa system claims to support this functionality [3]. Algorithms have animated to teach specific algorithms or data structures; Tango [12], GeoBench [10], Bals Zeus [2] are examples of animation environments which assist in this area.

The approach in this paper was designed with the intent of allowing students and scientists to learn and explore complex algorithms. Once the purpose of the animation targeted audience has been identified, the important features of the animation become a

- *Intuitive representations.* The choice of objects to represent the data structures is the understanding of the algorithm. Intuitive abstractions of the data aid in the und ing of the objects and their manipulation [6]. Also important is the transition o between successive frames in the animation [11].

- *User interaction.* Interaction is vital to learning [8]. Balsa-II [1] permits the mo of parameters at run-time to control aspects of the animation, while Zeus [2] all adjustment through direct manipulation and immediate update of all visible views

- *Animation controls.* Due to the diversity of skills and learning, the user should to control the speed of the animation and pause or stop at will. GeoBench [10] navigational controls such as stop, start, continue, automatic or step through a adjustment.

- *Multiple Views.* Illustrating different aspects of the algorithm simultaneously, code views for example, is a useful tool for displaying different aspects of an algorithm. alleviates the problem of trying to display too much detail on any one window.

- *History.* By storing an account of interactions and animations, a user can replay pa (rewind) or restart from a previous position in the animation. Balsa [3] stores such data in a script for such purposes. This is useful for re-exploring previous animatio

- *Granularity.* Being able to select the level of detail presented is vital to the unde of the algorithm. This is particularly true for complex algorithms. The ability to into a step to reveal and animate its substeps (expose more detail) allows for the and tailored exploration of the algorithm.

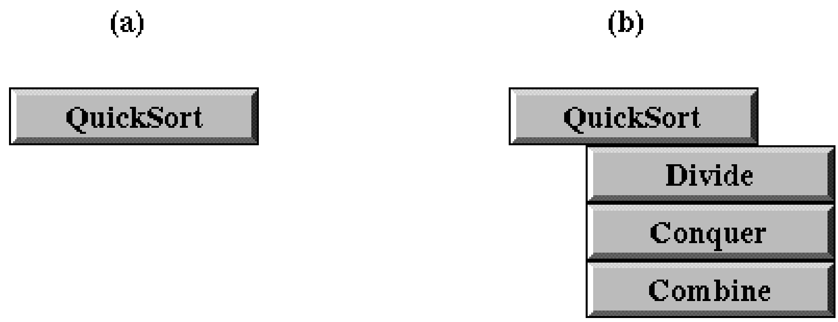

Figure 1: Expandable button example: (a) A collapsed button labeled "QuickSort". (b) "QuickSort" is to reveal its children (substeps): "Divide", "Conquer" and "Combine", which are collapsed.

- *Integration.* It is important that the student be able to relate each animation event in the algorithm, thus maintaining context. Balsa [3] meets this requirement by s view of the algorithm in textual form.

Given these requirements, the top-down approach is developed in the next section.

3 The Top-Down Approach

To create an algorithm animation suitable for top-down exploration, we propose an effec interface item called an *expandable button.* While an expandable button is similar in ap to a simple push button, its function is closer to that of a menu item. In some systems, a particular menu item reveals a submenu. Analogously, choosing an expandable butto multiple child buttons.

An expandable button is associated with every step in the algorithm and is labeled w scription of the step (see Figure 1). If its child buttons (corresponding to its substeps) ar then the button is *expanded*, otherwise it is *collapsed.*

To distinguish between different levels of abstraction, child buttons are placed directly b parent and their text indented to the right. Furthermore, collapsed buttons are colo expanded buttons are blue, and those without children are black.[1]

The expandable buttons are stored in a scrollable window; this facilitates algorithms too fit in a single window. Only those buttons displayed in the scrollable window will be an

[1]For these and subsequent figures, the colours of the text on the buttons reflect the conversion of colo scale, and do not represent highlighting.

Figure 2: Animation control panel.

The top-down algorithm animation proceeds as follows. Initially, there is only one butto
describing the name and purpose of the algorithm. When expanded, the highest level ste
algorithm are revealed.

The user selects *Play* from a control panel (Figure 2) to start the animation. The control p
vides other navigational controls for speed adjustments (pause time between display of a
frames), rewind, fast-forward, pause and stop.

In order to relate each selected step of the algorithm with the corresponding animation ev
expandable button has a *tag* associated with it. As each step is animated, its tag is hi
(see Figure 3).

The user has full control over which steps are animated and in how much detail, since o
corresponding to the visible expandable buttons are displayed. The user may choose to
the main steps to get an overall feel of how the algorithm works, then reveal more detai
more about about the intricacies of the algorithm. This allows users with different g
learning skills to view only those parts of the algorithm they are interested in.

This section has developed the concept of expandable buttons which when used in co
with an algorithm animation, allow for the top-down exploration of algorithms. To dem
the effectiveness of this approach, an example was created presenting a sophisticated alg

4 An Example

The top-down approach is particularly suitable for animating complex algorithms; in su
it is too overwhelming to present all the details at once. One such algorithm is the "
Planar Convex Hull Algorithm" developed by Kirkpatrick and Seidel [7].

Given a set of n points, the algorithm computes the H points ($H \leq n$) which lie on the b

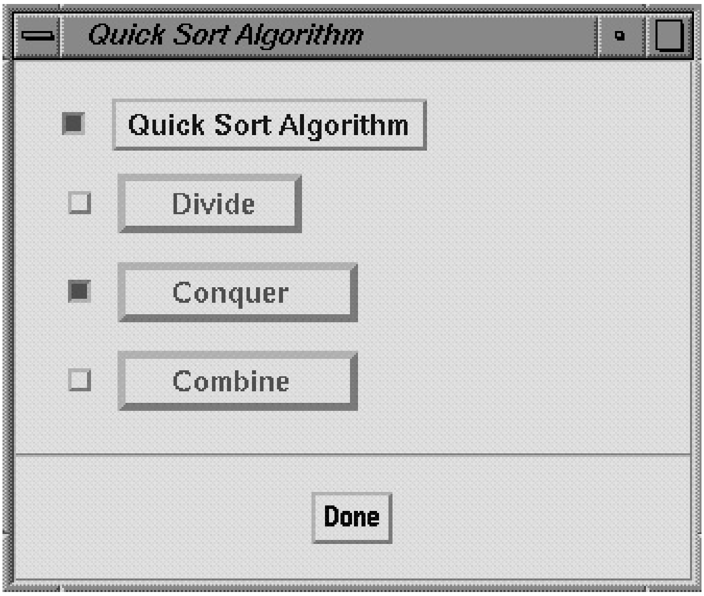

Figure 3: Associating algorithm steps to the animation via button tags. In this example, the animation
step *Conquer* within *Quick Sort Algorithm*.

of the convex hull of this set, with worst case time complexity $O(n\ log\ H)$. The convex h
smallest polygon which contains the entire set. Figure 4 shows the convex hull of a set o
Here the user has not expanded the initial "Ultimate Convex Hull Algorithm" button
animation consists of one image only displaying the convex hull.

The algorithm uses a variant of the divide and conquer paradigm called the *marria
conquest* principle. This principle breaks the problem into subproblems and determine
combine their solutions without actually computing them. This approach reverses the
steps of the traditional divide and conquer method of divide, conquer, and marry. The a
is that redundancies between subproblems to be merged can be eliminated.

The algorithm constructs the convex hull in two pieces, upper and lower, then merg
together. These three steps are displayed in Figure 5. Here the user expanded the
Convex Hull Algorithm button. The figure captures the animation of the computatio
upper hull.

The convex hull of the upper set of points is computed by dividing the set into two eq
halves using a vertical line and finding a *bridge*. The bridge is the edge of the convex h
intersects the dividing line. Points which lie below the bridge are eliminated since the
belong to the upper hull (see Figure 6). These steps are then applied recursively to th
remaining on either side of the vertical line (the two *ears*) to complete the upper hull.
illustrates the substeps resulting from the expansion of the *Compute the hull of the*
button. Note that these substeps are also expandable.

The lower hull is computed by reflecting the lower set of points about a horizontal line, tr
ing the set into an upper set and solving the problem as above. The result is then reflec
and merged with the upper hull to form the convex hull of the entire set.

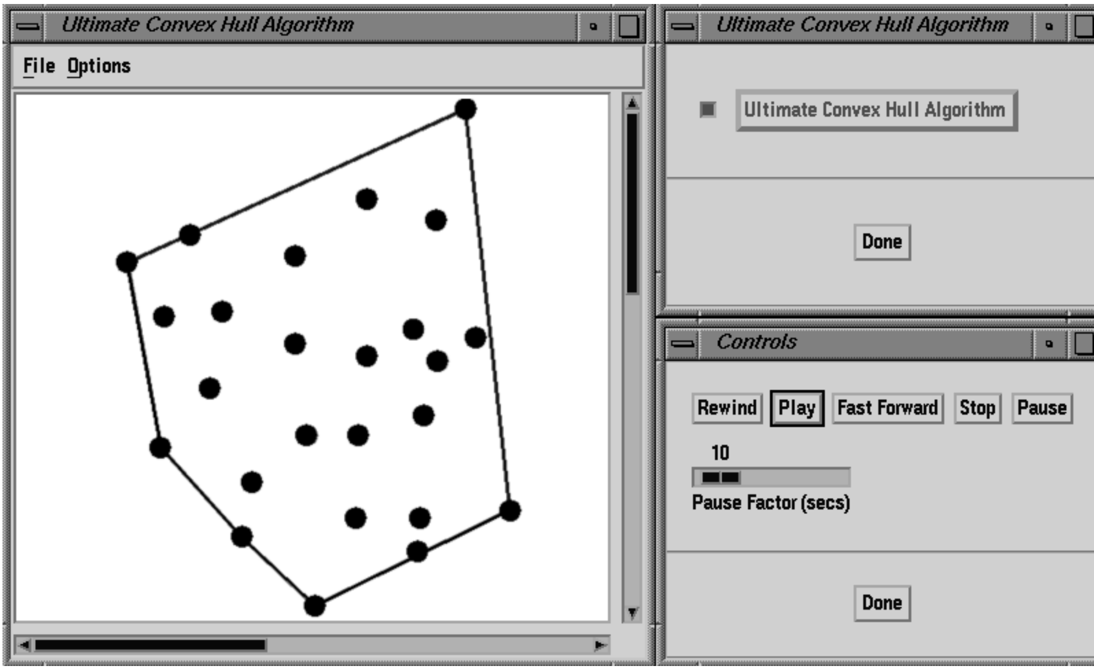

Figure 4: Animating the *Ultimate Convex Hull Algorithm* button displays the final output of the anin
convex hull of the points.

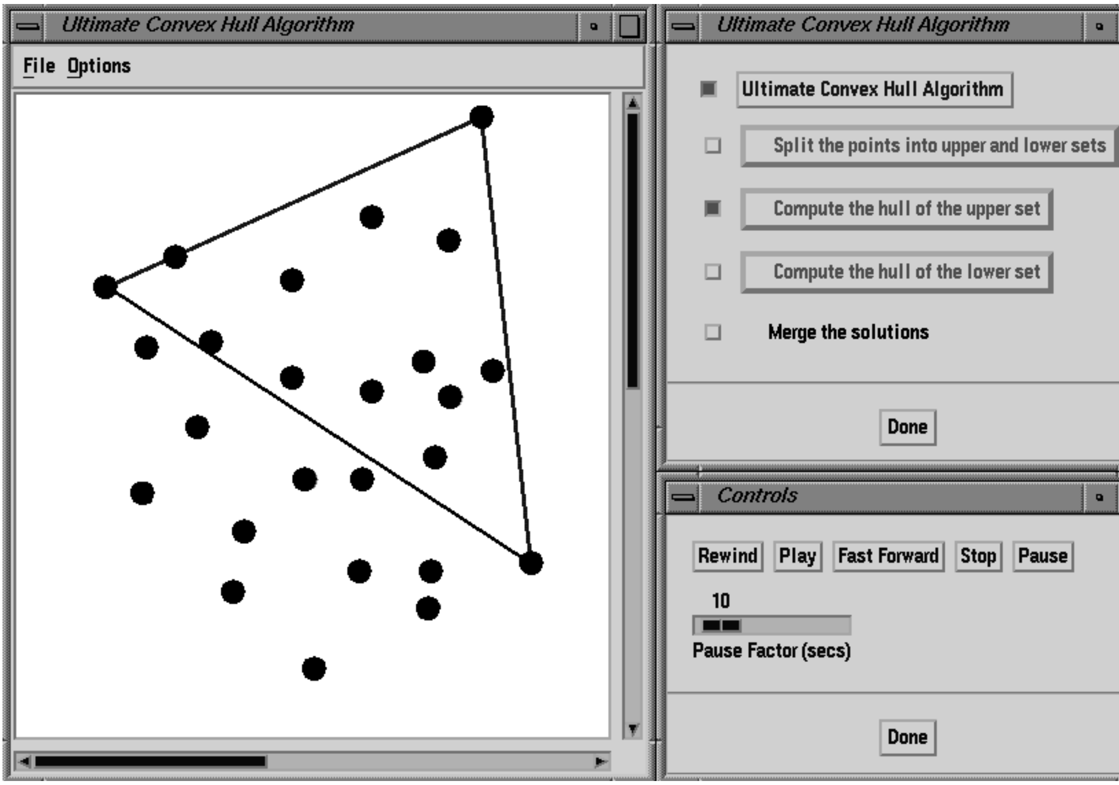

Figure 5: The main steps of the algorithm (the result of expanding the *Ultimate Convex Hull Algorith*
showing a frame of animation corresponding to the computation of the upper hull.

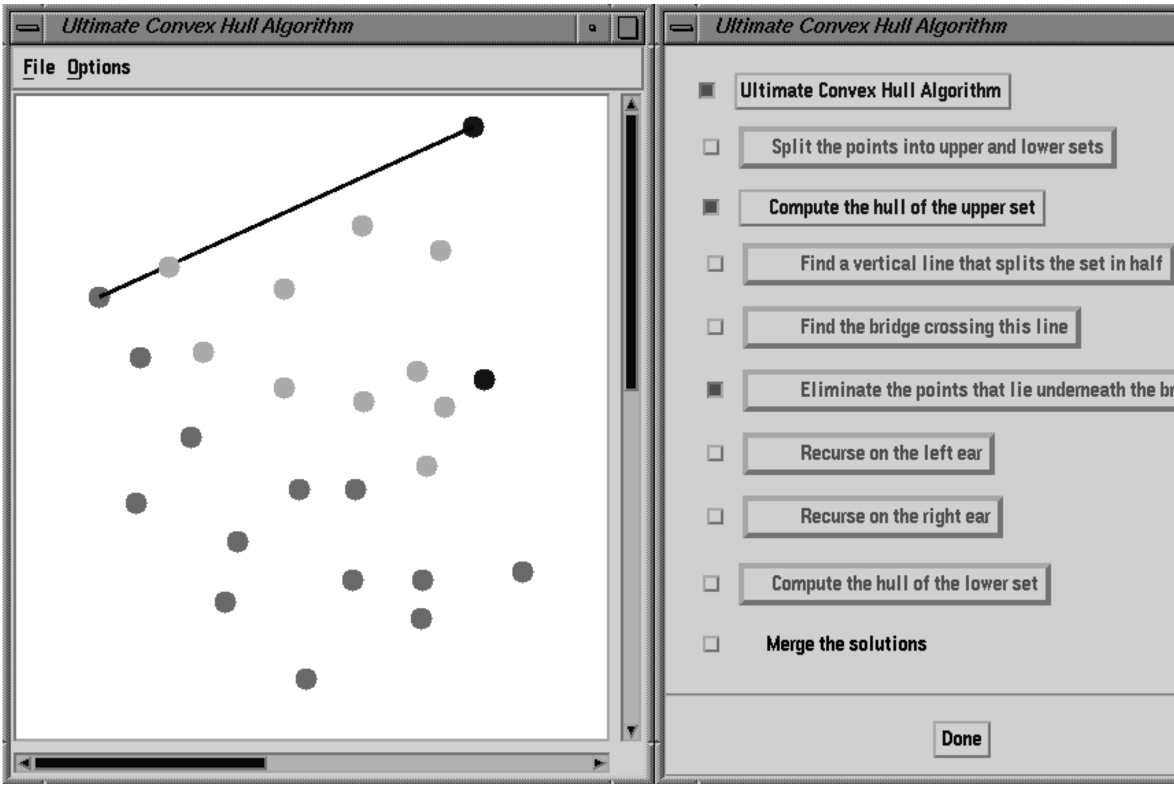

Figure 6: The granularity of the algorithm is altered by expanding the *Compute the hull of the upper set* animate the substeps involved in the computation of the upper hull. Points of the upper hull which are appear as light gray.

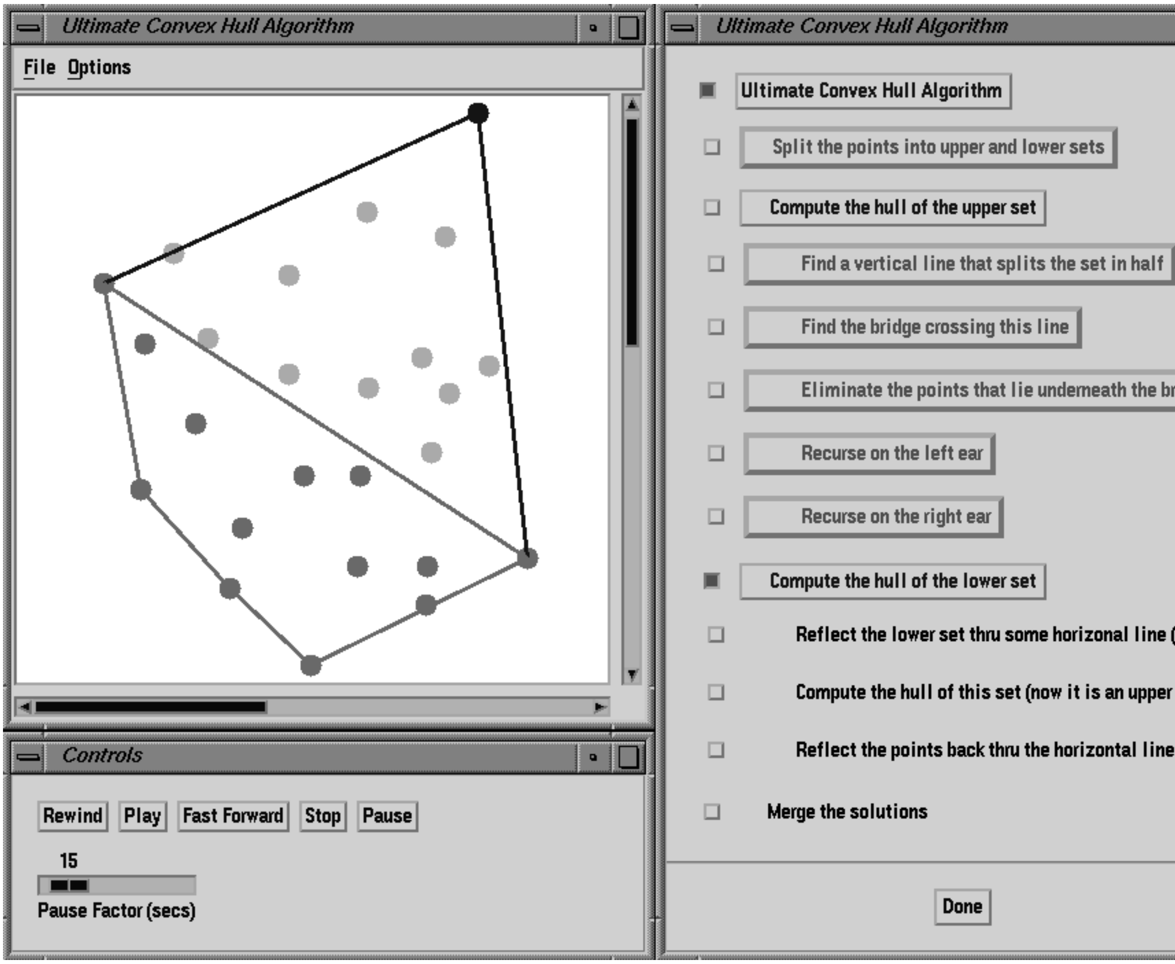

Figure 7: Further details as to the computation of the lower hull are selected (note that they do n further).

The algorithm animation, with varying levels of detail selected for each step in the co algorithm, is displayed in Figure 7. This example emphasizes the flexibility of the appro

5 Conclusions

This paper presents a novel interface approach for algorithm animation which utilizes ex buttons to allow for dynamic, top-down explorations of an algorithm. Traditionally, the t approach has been successful due to its ability to allow systematic abstractions of the it describes. In providing a flexible interface that reveals an algorithm in such a fashio benefits of this approach are integrated with those found in existing animations.

The advantages of this approach include:

- Varying levels of granularity are supported. The top-down approach reinforces st

thinking while allowing the user to personalize the amount of detail presented. Inf
can be hidden effectively to avoid obscuring or complicating the desired level of abs

- Algorithm pseudo-code and animation are integrated. Using the expandable bu display the algorithm in a simple structured format facilitates interaction with the a itself. This provides a verbal description of the algorithm animation as individual highlighted with their corresponding animation events. This association is vit learning process.

- Facilitates the animation design. From the animator's point of view, this approac suggestion of how to plan the presentation. By designing the animation from a t perspective, it is probable that the resulting animation has more clarity than one in an ad hoc fashion.

- Provides a mechanism for interface consistency. Since the algorithm is the user the method of interaction remains relatively similar for different algorithm animatio ensures that students can switch between different animations easily.

- Navigational controls are provided. In traditional teaching environments, the stu little or no control over the speed or order of the presentation. Allowing the student previous animations and control the pace of the motion further tailors the explo suit individual needs.

- Support for other requirements. Since the top-down approach does not interfere representation of the objects being animated, it supports the addition of further representations, interaction history, and multiple views.

It is worth noting that the top-down approach has some limitations. Not all algorithr top-down presentation, such as real-time, concurrent and object-oriented algorithms. F algorithms, it may be quite difficult to maintain context using the scrollable window alo

Since textual algorithm descriptions are integrated with the associated animations, it wo possible to extend the method for the purpose of program visualization. However, this is r since programs are usually much larger than their associated algorithms.

An advantage of using a top-down approach in systems programming, is that it pro avenue for reuse of algorithms and code. By allowing the expandable buttons to access animations, the approach provides a framework for integrating other algorithms.

In conclusion, the top-down approach provides an elegant solution to the problem of algorithms to students. The response to our example convinces us that the incorpo expandable buttons into an algorithm animation system would be beneficial.

Acknowledgements

The authors would like to thank Hausi Muller for presenting the convex hull algorithm down fashion so well, that it inspired this research. Thanks also to Brian Storey, Bryan

and Scott Tilley for suggestions that improved the readability of the paper. This work
supported by research and equipment grants from the Natural Sciences and Engineering
Council of Canada. We also gratefully acknowledge the support and facilities of the
University of Victoria and the Graphics and Multimedia Lab, School of Computing Scienc
Fraser University.

References

[1] M. H. Brown. Exploring algorithms using Balsa-II. *Computer*, May 1988.

[2] M. H. Brown. ZEUS: A system for algorithm animation and multi-view editing. In *Pr of the IEEE 1991 Workshop on Visual Languages, Kobe Japan*, pages 4–9, October

[3] M. H. Brown and R. Sedgewick. Techniques for algorithm animation. *IEEE Software*, 1985.

[4] M. H. Brown and R. Sedgewick. A system for algorithm animation. *ACM Computer* 18(3), July 1984.

[5] W. C. Bulgren, R. M. Marra, and G. F. Wetzel. An introductory algorithm teacher. *Bulletin*, 19(1):292–295, 1987.

[6] K. C. Cox and G.-C. Roman. Abstraction in algorithm animation. Technical Report 92-14, Washington University, St. Louis, March, 1992.

[7] D. G. Kirkpatrick and R. Seidel. The ultimate planar convex hull algorithm? *Comput.*, 15(1):287–299, February, 1986.

[8] B. R. Maxim and B. S. Elenbogen. Teaching programming algorithms aided by graphics. *SIGCSE Bulletin*, 19(1):297–301, 1987.

[9] G.-C. Roman and K. C. Cox. Program visualization: The art of mapping pro pictures. Technical Report WUCS-92-06, Washington University, St. Louis, Februa

[10] P. Schorn, A. Brungger, and M. de Lorenzi. The XYZ GeoBench: Animation of geo gorithms. In M. H. Brown and J. Hershberger, editors, *Animations for Geometric Al A Video Review*, Palo Alto, California, 1992. Digital Systems Research Center.

[11] J. T. Stasko. The path-transition paradigm: A practical methodology for adding a to program interfaces. *Journal of Visual Languages and Computing*, 1:213–236, 19

[12] J. T. Stasko. Tango: A framework and system for algorithm animation. *IEEE C* September, 1990.