August 2001

Visualization as a key element in learning

J. Fulcher
University of Wollongong, john@uow.edu.au

Visualization as a key element in learning

Abstract

Experience with teaching undergraduate and postgraduate computer science (CS) subjects has shown that students benefit substantially from being able to visualize fundamental topics. Examples are presented from five different CS areas, namely: (i) communications, (ii) parallel computing, (iii) artificial neural networks, (iv) genetic algorithms and (v) data mining. The majority of these examples stemmed from student projects, rather than commercial software products. Accordingly, students benefit twofold from the visually-oriented subject presentations, firstly in facilitating the learning of basic concepts, and secondly in the development of educational software packages as part of their project work.

Disciplines

Physical Sciences and Mathematics

Publication Details

This paper originally appeared as: Fulcher, JA, Visualization as a key element in learning, Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology, 19-22 August 2001, vol 1, 469-472. Copyright IEEE 2001.

Visualization as a Key Element in Learning

John A. Fulcher, ***Senior Member, IEEE***

***Abstract* -- Experience with teaching undergraduate and postgraduate Computer Science subjects has shown that students benefit substantially from being able to visualize fundamental topics. Examples are presented from five different CS areas, namely (i) communications, (ii) parallel computing, (iii) Artificial Neural Networks, (iv) Genetic Algorithms and (v) Data Mining. The majority of these examples stemmed from student projects, rather than commercial software products. Accordingly, students benefit twofold from the visually-oriented subject presentations, *firstly* in facilitating the learning of basic concepts, and *secondly* in the development of educational software packages as part of their project work.**

***Index Terms* — Case studies, computer science, pedagogy, simulation, teaching methods.**

I. INTRODUCTION

IN the Preface of my 1989 textbook [1], I stated that "I have always been a firm believer in the proverb 'one picture is worth more than a thousand words'. Accordingly, I have deliberately made *extensive* use of diagrams throughout this text. It is my belief that *seeing* how a particular peripheral device works is half the battle in interfacing it to a computer." Nothing has occurred during the intervening years to alter my view on this matter.

The aim of this paper is to present a few case studies which illustrate the value of adopting a visual approach to the teaching of specific topics from both undergraduate and postgraduate Computer Science subjects at the University of Wollongong.

II. COMMUNICATIONS PROTOCOLS

The field of telecommunications is filled with acronyms – indeed, to the novice (e.g. our undergraduate students meeting the field for the first time), it can appear that practitioners are speaking a foreign language! Nowhere is this more so than with communications protocols. Accordingly, a project was undertaken to develop a visual teaching tool to assist in the teaching of both the International Standards Organization 7-layer Open Systems Architecture model and the ubiquitous Transmission Control Protocol/Internet Protocol suite.

Being able to visualize the generation of packets at each different layer of OSI and TCP/IP greatly facilitates students' understanding of how information is decomposed at the transmitting station (node), sent over the communications channel, then re-assembled at the distant receiver. An audiovisual demonstration of this process was developed on a Silicon Graphics O2 graphics workstation, using both `OpenGL` and SGI's `OpenInventor`.

III. PARALLEL COMPUTING

Following on from the success of the communication protocols project (Section II), it was decided to extend this idea to a more general `OpenGL` platform, this time with a postgraduate subject in parallel programming as the target. Now SGI used *extensions* to the `OpenGL` standard in their implementation of `MineSet` (see Section VI below). We were keen to develop a general `OpenGL` visualization tool which would facilitate porting across platforms. For this reason, we targeted (Wintel) PCs.

Figure 1 shows a snapshot from the Parallel Computing Teaching Tool – in this instance, an animation introducing Interconnection Networks.

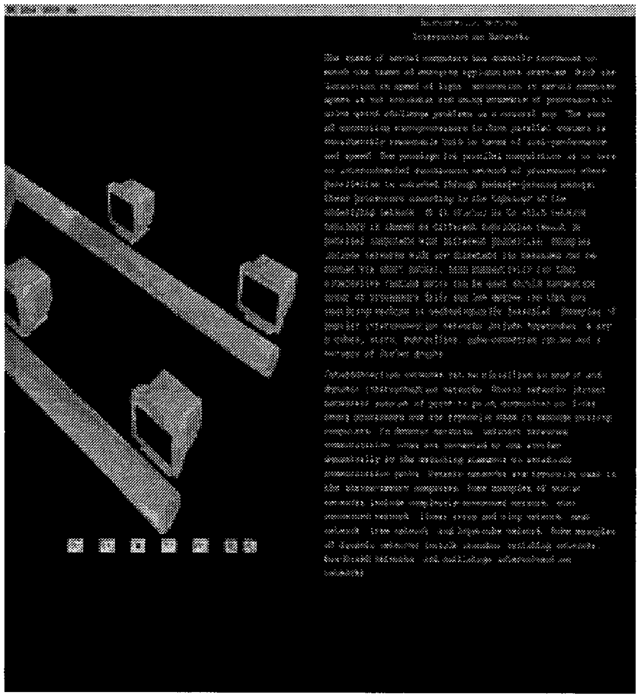

Fig. 1. Snapshot from Parallel Computing Teaching Tool

This work was supported in part by (i) an AspenTech (NeuralWare) University Teaching Grant, (ii) a Research Infrastructure Block Grant (SGI Origin 200) from the Australian Research Council, and (iii) the Institute for Mathematical & Computer Modelling at the University of Wollongong (Intelligent Systems Research Group).

John. A. Fulcher is with the School of Information Technology and Computer Science, University of Wollongong, Australia (telephone: +61 242 213811, e-mail: john@uow.edu.au).

** LEAVE TWO LINES HERE FOR IEEE TO ADD THE PUBLICATION NUMBER AND OTHER CONFERENCE INFORMATION.

IV. TRAINING IN ARTIFICIAL NEURAL NETWORKS

BackPropagation learning in Multi-Layer Perceptrons can be viewed as gradient descent in weight space. Visualizing such learning in (simple) energy landscapes greatly assists students in their understanding of Artificial Neural Networks. To assist in this process, `MatLab` demonstrations are incorporated into the lecture presentations in the graduate ANN subject.

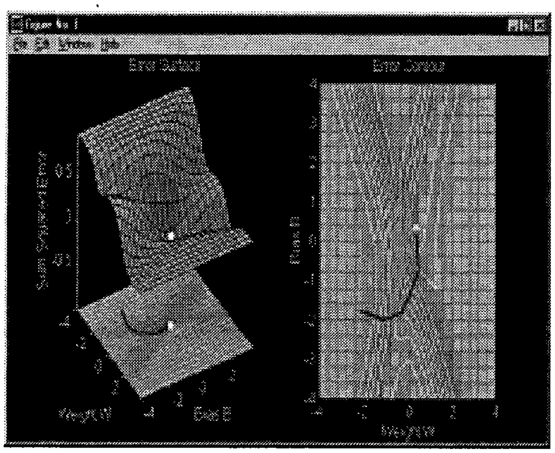

Fig. 2. BackPropagation learning (Gradient Descent in Weight Space).

In the simple 3D energy landscape of Figure 2, the overall network error is plotted as a function of the network weights (only two in this example - one of which is the threshold or bias weight). Ordinarily such low-dimensional energy landscapes would rarely be encountered in real-world applications, however by visualizing such simple landscapes, students gain an appreciation of what is taking place in higher-dimensional "hills" and "valleys". Network training corresponds to movement towards the global minimum, plotted as both 3-Dimensional (left) and 2D plan (right) views in Figure 2.

Figure 3 shows the use of ANNs to recognize printed characters. The network had been previously trained using characters such as the upper and lower ones in Figure 4. In this instance, the addition of around 50% noise causes the network to begin to recall *incorrect* characters.

As well as incorporating MatLab demonstrations into the lecture presentations, students use `NeuralWorks Professional-II+` in their laboratory assignment work. It should be emphasized that the latter is used *in preference to* later product offerings from NeuralWare Inc. (such as `NeuroSIM`), due to the visualization capabilities of the former software package [2-6].

For example, Figure 4 shows a Standard BackPropagation network (Multi-Layer Perceptron) learning the eXclusive-OR function (which it accomplishes after 500 epcohs or so – an epoch being one complete pass through the input-output training data). Being able to dynamically display network characteristics greatly assists students in gaining an understanding of network training.

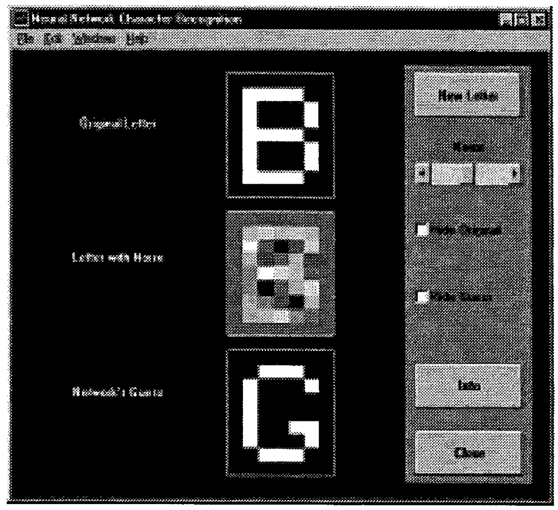

Fig. 3. Application of ANNs to Printed Character Recognition.

In Figure 4, overall network error is plotted as a function of time (epoch), together with (positive or negative) network weights and (actual versus desired) classification rate (perfect recall in this case).

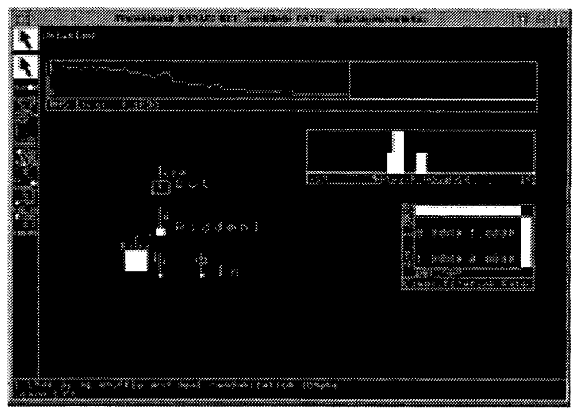

Fig. 4. Learning XOR in a Multi-Layer Perceptron (BackPropagation).

V. EVOLUTION OF FACES USING GENETIC ALGORITHMS

Our initial interest in face recognition was sparked by an airport security project undertaken during the early 1990s [7]. Genetic Algorithms/Evolutionary Computing is also covered in the ANN graduate subject referred to in Section IV.

As part of their undergraduate degree studies, Computer Science students are required to work on a group software

project (comprising between 4 and 6 members). This project forms the capstone of their previous two years studies, and extends over the final (third) year of their Bachelor of Computer Science. Usually the project topic is driven by the research interests of the CS faculty, especially if these overlap with industry.

In the present case, a final-year software group project was undertaken in the evolution (morphing) of faces using Interactive Genetic Algorithms. By interactive, we mean that the usual evolutionary process (mutate Ł select_parents Ł reproduce_offspring Ł kill_off_unfit_children Ł mutate…) is halted after each new generation, to enable the user to direct the next evolutionary step. They perform this "filtering", guidance or direction by simply clicking the mouse on the "best match" from amongst the 13 faces presented on the computer screen, as illustrated in Figure 5. Thirteen further faces are then evolved, and the entire process is repeated until the user is satisfied with one of the 13 "final" offerings.

Fig. 5. Selection of Evolved Faces.

Initially it was thought that such a facility would be of use to law enforcement agencies, however it subsequently proved unsuitable, since its operation could be construed as "leading the witness" from a legal viewpoint (e.g. matching of perpetrator's faces from mug shots). As it transpired, the target area of application was more likely to be in the medical/dental field(s) (e.g. the effects of aging/plastic surgery, facial reconstruction and the like).

Faces are evolved from an initial description – comprising eyes, nose, mouth, ears and hair. Faces were obtained from the http://www.geocities.com/CapeCanaveral/1624/ site.

Essentially this project is entirely visually-driven, at least from the user's perspective. It was implemented in C++ on Unix (SolarisV8). The visual nature of this group software project helped the students develop a product which was very much "customer-focused".

VI. VISUALIZATION OF DATA MINING

1999 saw the first offering within the Faculty of Informatics at the University of Wollongong of a graduate subject on data mining. This subject is co-taught by Computer Science and Statistics. On the Computer Science side, we bring both database and AI/machine learning approaches to data mining (OLAP, association rules and rough sets in the case of the former; decision trees, artificial neural networks and genetic algorithms in the latter).

Because we had access to Silicon Graphics machines, we decided to adopt SGI's `MineSet` tool for this new subject offering. Whilst `MineSet` does not offer any AI features, it does incorporate the following tools:

(i) Decision/Option/Evidence Tree/ Table Inducers & Classifiers,
(ii) Clustering algorithm,
(iii) Regression Tree, and
(iv) Column Importance.

Perhaps its *major* feature however is its visualization tools (scatter/splat, histogram). Figure 6 (3.7) is taken from a `MineSet` Splat Visualizer window. In this display, the independent data columns are mapped to the x-y-z axes (total day charge, number of customer service calls and international plan, respectively).

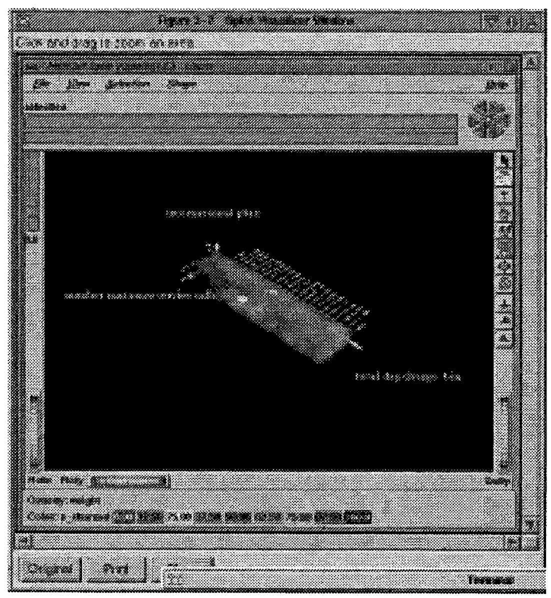

Fig. 6. Splat Plot from MineSet Churn Tutorial.

It is a little difficult to appreciate from Figure 6, but on a colour graphics terminal the data clusters become immediately apparent (0.00 = blue Ł 100.0 = red). In this particular example, there are two clusters of significance, namely (i) lower right, which corresponds to high total day charge, and

(ii) upper left, which corresponds to low total day charge and high customer service calls.

This figure is taken from `MineSet`'s `churn` tutorial, so named because of the effect of customers changing their preferred telecommunications carrier. Telephone companies hold databases of calls, billing information, customers and customer service, which together constitute customer "signatures". Obviously what the companies are interested in is explaining why customers "churn"

After identifying three (k-means) clusters - blue, red and green - the column importance tool within `MineSet` was activated. The three most significant independent variables were then assigned to the x-y-z axes, these being the evening call time, day call time and number of voicemail messages, respectively. The "blue" (front/lower right) cluster in the scatter plot of Figure 7 (4.6) corresponds to cluster#2. By contrast, the flat pancake shape (representing low numbers of voicemail messages) is distributed evenly between the "red" and "green" clusters. We can see immediately that the total numbers of daytime and evening minutes are interdependent.

Fig. 7. Scatter Plot from MineSet Churn Tutorial.

The value of displaying data in such visual manner(s) is twofold – *firstly* data clusters become readily apparent, and *secondly* unexpected "gems" can become immediately obvious (the whole aim of data mining after all is to extract non-obvious/unexpected "gems" from the predominately "worthless" topsoil).

VII. Conclusion

The five examples presented above all highlight the usefulness of visualization from a teaching perspective. Our experience has been that students benefit greatly from having software demonstrations available at their fingertips which illustrate the underlying principles of the topics under study.

References

[1] J.A. Fulcher, *An Introduction to Microcomputer Systems: Architecture and Interfacing*. Reading, MA: Addison Wesley, 1989.

[2] J.A. Fulcher, "Teaching Neural Networks" in *Computers in Education: Proc. 5th IFIP World Conf. Computers in Education – WCCE'90*, Sydney, 9-13 July, A. McDougall and C. Dowling, Eds. North Holland: Elsevier, 1990, pp.275-280.

[3] J.A. Fulcher, "Experience with Teaching a Graduate Neural Networks Course," *Computer Science Education*, vol.3, no.3, pp.297-314, 1992..

[4] J.A. Fulcher, "Neural Network Alternatives," *Proc. 6th IFIP World Conf. Computers in Education – WCCE'95*, Birmingham, UK, 23-28 July, p.324, 1995.

[5] J.A. Fulcher, "Laboratory Support for the Teaching of Neural Networks," *Intl. J. Electrical Engineering Education.*, vol.35, no.1, pp.29-36, 1998.

[6] J.A. Fulcher, "Teaching Course on Artificial Neural Networks", in *Innovative Teaching Methods in Artificial Intelligence, Neural Networks, Fuzzy Systems and Evolutionary Computing*, L. Jain, Ed., Berlin: Physica Verlag, 2000.

[7] M. Zhang and J.A. Fulcher, "Face Recognition Using Artificial Neural Network Group-Based Adaptive Tolerance Trees", *IEEE Trans. Neural Networks*, vol.7, no.3, pp.555-567, 1996.