Visualizing Real-Time Multivariate Data Using Preattentive Processing

CHRISTOPHER G. HEALEY, KELLOGG S. BOOTH, and JAMES T. ENNS
The University of British Columbia

A new method is presented for visualizing data as they are generated from real-time applications. These techniques allow viewers to perform simple data analysis tasks such as detection of data groups and boundaries, target detection, and estimation. The goal is to do this rapidly and accurately on a dynamic sequence of data frames. Our techniques take advantage of an ability of the human visual system called preattentive processing. Preattentive processing refers to an initial organization of the visual system based on operations believed to be rapid, automatic, and spatially parallel. Examples of visual features that can be detected in this way include hue, orientation, intensity, size, curvature, and line length. We believe that studies from preattentive processing should be used to assist in the design of visualization tools, especially those for which high speed target, boundary, and region detection are important.

Previous work has shown that results from research in preattentive processing can be used to build visualization tools which allow rapid and accurate analysis of individual, static data frames. We extend these techniques to a dynamic real-time environment. This allows users to perform similar tasks on dynamic sequences of frames, exactly like those generated by real-time systems such as visual interactive simulation. We studied two known preattentive features, hue and curvature. The primary question investigated was whether rapid and accurate target and boundary detection in dynamic sequences is possible using these features. Behavioral experiments were run that simulated displays from our preattentive visualization tools. Analysis of the results of the experiments showed that rapid and accurate target and boundary detection is possible with both hue and curvature. A second question, whether interactions occur between the two features in a real-time environment, was answered positively. This suggests that these and perhaps other visual features can be used to create visualization tools that allow high-speed multidimensional data analysis for use in real-time applications. It also shows that care must be taken in the assignment of data elements to preattentive features to avoid creating certain visual interference effects.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces–*ergonomics, screen design (graphics, colour)*; I.3.6 [**Computer Graphics**]: Methodology and Techniques–*ergonomics, interaction techniques*; I.6.6 [**Simulation and Modeling**]: Simulation Output Analysis

General Terms: Design, Experimentation, Performance

Additional Key Words and Phrases: boundary detection, cognitive psychology, curvature, icon, hue, human vision, multivariate data, preattentive, scientific visualization, target detection, visual interactive simulation

1. INTRODUCTION

The field of scientific visualization draws on research from a wide spectrum of traditional disciplines. These include computer science, psychology, and the visual arts. The "domain of visualization", as defined by a National Science Foundation panel on scientific computing, includes the development of specific applications, the development of general purpose tools, and the study of research problems that arise in the process [McCormick et al., 1987; Rosenblum, 1994]. To date, most research efforts have focused on ad hoc visualization applications. Relatively few efforts have formulated general guidelines for the design of visualization tools.

In this paper, we report on new work that derives from an area of cognitive psychology known as preattentive processing. This work is part of an on-going investigation whose goal is a set of guidelines for visualization design.

Authors' addresses: C.G. Healey and K.S. Booth, Imager Computer Graphics Laboratory, Department of Computer Science, 2366 Main Mall, University of British Columbia, Vancouver, British Columbia, V6T 1Z4, Canada; healey@cs.ubc.ca, ksbooth@cs.ubc.ca. James T. Enns, Department of Psychology, 2136 Main Mall, University of British Columbia, Vancouver, British Columbia, V6T 1Z4, Canada; jenns@cortex.psych.ubc.ca.

We examine visualization techniques for dynamic sequences of multidimensional data frames, like those produced by visual interactive simulation or real-time systems. We first define a set of visualization requirements that are common to these applications. We then review research in preattentive processing in order to establish abilities and limitations of human vision that are relevant to these requirements, after which we describe a scientific visualization tool we have developed that is based on these general considerations. Finally, we discuss the implications of our approach, both for specific applications and for the development of general guidelines in scientific visualization.

2. VISUALIZATION REQUIREMENTS

Scientific visualization as a discipline within computer graphics is a relatively recent development. The first reference to "scientific visualization" *per se* occurred sometime in the late 1980s, although as [Fournier, 1994] has pointed out, many aspects of scientific visualization have long been part of computer graphics. Panels and workshops in a variety of different disciplines are now addressing scientific visualization and its relationship to their work [Becker and Cleveland, 1991; Rosenblum, 1994; Treinish et al., 1988; Wolfe and Franzel, 1988]. The area is expanding into a number of subfields that use computer graphics to solve various types of problems. Examples include volume visualization, medical imaging, flow visualization, and multivariate data visualization. Current research is concerned with the design of intelligent visualization tools. Scientists are turning to computer graphics, psychology, and visual arts to understand how the human visual system analyses images. This has led to the use of specific visual properties to make displays more intuitive. These visual properties take advantage of the fundamental workings of the visual system itself.

Many applications require techniques for displaying data in real-time. One example is air traffic control, where displays are often shared by different operators, who acquire visual data from different parts of the display at the same time. The visualization technique must allow a variety of tasks to be performed rapidly and accurately on dynamically changing subsets of the overall display. Medical imaging systems such as CT, MRI, PET, and ultrasound are another type of application that could benefit from real-time visualization. A method that allowed rapid and accurate visual analysis of more than one aspect of the data would decrease the amount of time needed to complete the diagnostic task. This is important, because these types of systems often cannot be time-shared by multiple users. Any improvement in visualization would increase total throughput for the system.

2.1 Simulation Visualization Systems

The requirements for real-time applications are similar to another class of problems, the visualization of output from simulation systems. Disciplines such as physics, chemistry, oceanography, and management science use simulations to model real-world phenomena. Discrete event simulation systems have matured far beyond their original beginnings as languages like GPSS, Simula, and SIMSCRIPT. Advances in computer processing power and display capabilities have been matched by simulation systems which offer graphical network builders, user interaction, and real-time visualization of results as they are generated. These systems are designed to address a wide range of problem environments. Some languages have been optimized to model specific applications (*e.g.* manufacturing operations). Others target a more general class of problems, providing the flexibility to build a wide variety of simulations through the use of graphical model builders and system-specific languages. Our interest is in the systems used to animate simulation results. Brief descriptions of Proof, Cinema, Arena, SIMGRAPHICS, and SLAMSYSTEM are provided below, highlighting the various capabilities of each of these packages.

Proof is a PC-based animation system which displays results from languages like GPSS and its successors, GPSS/H and SLX [Earle et al., 1990; Henriksen, 1993]. It provides a number of simple graphics and animation primitives. Proof runs as a post-processor, separating the simulation model from the animation system. This means the simulation model can be executed in a more powerful computing environment (*e.g.* on a mainframe or a UNIX workstation) if necessary. Results can then be downloaded to the PC for viewing. Post-processing ensures the speed of the animation is not limited by the speed of the underlying simulation. Unfortunately, it also makes it impossible to interactively change values as the simulation runs and watch the corresponding impact.

Cinema is a general-purpose animation system built on top of the SIMAN simulation language [Kalasky, 1991]. Cinema displays data in real-time as the simulation executes. Users can temporarily halt the simulation and use SIMAN's interactive commands to view or change the state of the system being modeled. Cinema's creators suggest using animation to enhance four stages of the design process: model building and verification, model validation, bottleneck analysis, and communication and presentation.

SIMAN and Cinema have also been combined into a hierarchical modeling and animation system called Arena [Collins and Watson, 1993]. Arena models are built using templates and modules. A "SIMAN template" is included as a standard part of the Arena system. The template contains modules for each of the basic modeling elements in SIMAN (*e.g.* Queues, Variables, and Transporters). A different template contains another set of modules representing basic Cinema animation elements. Users build new modules by combining already-existing ones in various ways. These "derived modules" usually correspond to entities specific to the application being modeled (*e.g.* jobsteps and production schedules in a wafer fabrication template). Once the modules and templates are designed, Arena's interface can be used to build, run, interrupt, inspect, and modify the simulation.

SIMSCRIPT II.5 is another well known simulation language. Like its competitors, it allows the user to interactively debug and modify models as they run [Garrison, 1990]. SIMGRAPHICS is provided to allow the user to display results graphically. Users can choose from a number of standard presentation graphics like pie charts, bar charts, and level meters. It is also possible to animate icons over a static background. Graphics are drawn concurrently as the simulation runs.

Finally, SLAM II is a simulation language designed to support models built using a combination of the process flow, discrete event, and continuous time concepts [O'Reilly and Whitford, 1990]. SLAMSYSTEM is the PC-based version of SLAM II [Lilegdon and Ehrlich, 1990]. It uses the display capabilities of the PC to provide a graphical network builder, presentation graphics, and animation. SLAMSYSTEM can display results in real-time as the simulation runs, or in a post-processing fashion. This provides the flexibility of either interactive visualization of results, or separate computing and animation environments.

2.2 Visual Interactive Simulation

The animation systems discussed above were designed to address a number of drawbacks inherent in first-generation simulation languages like GPSS, Simula, and SIMSCRIPT. Descriptions of these systems show many of them share a number of common goals:

- the ability to analyse results graphically, rather than simply receiving a set of numbers when the simulation system terminates
- the ability to see intermediate results while the simulation is running in order to verify that the system is executing properly, or to understand why and how final results are being produced
- the ability to interact with the simulation while it runs, guiding it to follow interesting results or trends as they occur; this also avoids the difficult problem of trying to recreate a given system state by simply modifying the initial starting conditions

Visual interactive simulation (VIS) is a technique specifically designed to study the above problems [Hurrion, 1976]. It consists of two parts: visualization of simulation data and user interaction with a running simulation [Bell and O'Keefe, 1987]. After building a number of models for various simulation software components, [Hurrion, 1980] offered a set of anecdotal observations supporting the use of VIS. He suggested that VIS made users more attentive, partly because pictures were more interesting than text and partly because users felt they had more control over the simulation system. Visualization of intermediate results sometimes gave rise to interesting situations the user had never envisioned.

Bell and O'Keefe have recently addressed the argument that VIS is a method for solving problems with simulation models, rather than a technique for building the models themselves [Bell and O'Keefe, 1994]. They note that every VIS system contains two distinct parts, a simulation model and a visualization model. From this staring point, Bell and O'Keefe divide VIS into two categories: active and passive.

In active VIS, there is a one-to-one mapping between elements in the simulation model and elements in the visualization model. Users can "see" the entire simulation. This means the user can take an active role in defining and understanding the simulation model, and in suggesting alternatives. Bell and O'Keefe classify active VIS tools as a type of decision support system. Passive VIS, on the other hand, separates the simulation and visualization models. A modeling expert builds the simulation model, then decides which parts to animate for the user. Control over problem solving is assigned to the model builder, rather than the user. Passive VIS is simply traditional simulation with the addition of animation.

There are a number of reports showing practical examples of VIS applications. [Kaufman and Hanani, 1981] described a method of converting a traditional simulation program into an interactive program that used graphics. Researchers in the United Kingdom developed a set of library routines that perform VIS on an Apple microcomputer [O'Keefe and Davies, 1986]. After applying their software to various practical simulation problems, they concluded that a machine with relatively simple graphics primitives and low computational power offered an acceptable platform for VIS software. Scientists at AT&T Bell Labs have developed a general VIS package, the Performance Analysis Workstation [Melamed and Morris, 1985], that allows design and testing of queueing networks. The simulation can be started, interrupted, modified, and monitored using commands available through pop-up menus. The workstation visually displays activity in the network, including movement and queueing of requests as the simulation executes. Experience using this workstation mirrors observations of Hurrion, specifically that users enjoy the workstation and that interesting and unexpected phenomena often arise while the simulation is running.

GENETIK and WITNESS are two recent applications designed to perform active VIS. Both are loosely based on SEE-WHY [Fiddy et al., 1981], one of the first commercial VIS packages. GENETIK allows users to build and execute general purpose simulation models [Concannon and Becker, 1990]. Its core system comprises four units: *graphical units* which display and animate result, *logic units* which define the behaviour of the simulation model, *data units* which represent local and global variables, and *interaction units* which allow users to interact with the simulation model. GENETIK provides a "simulation module" as a basic framework for building discrete-event and continuous simulation models. The prepackaged pieces available in the simulation module can be combined with additional graphic, logic, data, or interaction units as required to build a complete simulation model. GENETIK also provides a "planning board module" to help design scheduling simulations, essentially through the addition of Gantt charts.

WITNESS is a successor to the SEE-WHY simulation system. It was specifically designed to analyse manufacturing systems [Murgiano, 1990; Clark, 1991]. The most obvious addition is an interactive model building environment. Unlike SEE-WHY, which used FORTRAN or a built-in simulation language, WITNESS models can be created through the use of menus and input prompts. WITNESS runs in an interpreted fashion, which means partially completed models can be executed to validate their design. Since it is a VIS tool, WITNESS has a complete set of functions to interrupt and modify simulations as they run. Results are displayed as a combination of static icons and system-driven animations.

2.3 Real-Time Multivariate Data Visualization

This examination of available visualization and simulation animation systems shows we now have the computing power and display capabilities to represent information visually in many different ways. Unfortunately, there is no guarantee that an ad hoc set of displays will help us better understand our data. Researchers are working to find ways to harness these resources through the use of effective and efficient visualization techniques.

An intuitive and often used approach is to associate "features" such as hue, spatial location, and size with each data element. These features represent specific attributes embedded in the element (*e.g.* colour is often used to represent temperature at each spatial location on a map). This addresses an explicit goal of visualization, to present data to the human observer in a way that is informative and meaningful, on the one hand, and yet intuitive and effortless on the other. Features are chosen to show properties within and relationships among data elements. Unfortunately, an arbitrary assignment of features to individual data dimensions may not result in a useful visualization tool. A poor choice can lead to a tool which actively interferes with a user's ability to extract the desired information.

Several researchers have suggested harnessing the human visual system to assist with this problem. For example, [Enns, 1990] discusses using the human visual system to process large datasets efficiently; he describes geometric icons which combine the power of the computer and the human visual system [Enns and Rensink, 1990]. Ware and Beatty have designed a method that uses colour to represent multidimensional data elements [Ware and Beatty, 1988]; subsets of the data with similar values appear as a spatial "cloud" of similarly coloured squares. Pickett and Grinstein [1988, 1989] display structure in the data as a set of textures and boundaries, so that groups of data elements with similar values appear in the display as a spatial group with a unique texture. Previous work in our laboratory has shown that preattentive features can be used for estimation [Healey et al., 1993; Healey et al., 1994]; subjects are able to rapidly and accurately estimate the relative percentage of data elements in the display with a specific preattentive feature. These techniques were all applied to a single data frame in isolation. Aside from Pickett and Grinstein, none have modified their technique to display multiple data frames one after another in an animated fashion.

We approached real-time multivariate visualization by defining a set of requirements which we feel are inherent to this class of problem:

- *multidimensional data:* the technique should be able to display multidimensional data in a two-dimensional environment, the computer screen
- *shared data:* the technique should display independent data values simultaneously; a single user could choose to examine various relationships, or multiple users could simultaneously examine independent data values
- *real-time data:* the technique must function in a real-time environment, where frames of data are continuously generated and displayed one after another

- *speed:* the technique should allow users to rapidly obtain useful and nontrivial information; here, "rapidly" means less than 250 milliseconds (msec) per data frame
- *accuracy:* information obtained by the users should accurately represent the relationship being investigated

Using an approach which extends our previous work on static visualization, we decided to use preattentive processing to assist with real-time (dynamic) multivariate data visualization. We hypothesized that important aspects of preattentive processing will extend to a real-time environment. In particular, we believe real-time visualization techniques based on preattentive processing will satisfy the five requirements listed above. A visualization tool which uses preattentive features will allow viewers to perform rapid and accurate visual tasks such as grouping of similar data elements (boundary detection), detection of elements with a unique characteristic (target detection), and estimation of the number of elements with a given value or range of values, all in real-time on temporally animated data frames. We tested this hypothesis using behavioral experiments that simulated our preattentive visualization tools. Analysis of the experimental results supported our hypothesis for boundary and target detection. Moreover, interference properties previously reported for static preattentive visualization were found to apply to a dynamic environment.

3. PREATTENTIVE PROCESSING

Researchers in psychology and vision have discovered a number of visual properties that are "preattentively" processed. They are detected immediately by the visual system. This means viewers do not have to focus their attention on a specific region in an image to determine whether elements with the given property are present or absent.

An example of a preattentive task is detecting a filled circle in a group of empty circles (Figure 1a). A viewer can quickly glance at the image to determine whether the target is present or absent. Commonly used preattentive features include hue, curvature, size, intensity, orientation, length, motion, and depth of field. As mentioned above, simply choosing features in an ad hoc manner and matching them to data attributes will not necessarily result in an intuitive display. A "conjunction" occurs when the target object is made up of two or more features, each of which is contained in the distractor objects. Objects that are made up of a conjunction of unique features cannot be detected preattentively [Triesman, 1985]. Figure 1b shows an example of a conjunction task. The target is made up of two features, filled and circular. Both these features occur in the distractor objects (filled squares and empty circles). Thus, the target cannot be preattentively detected.

Properties that are processed preattentively can be used to highlight important image characteristics. Experiments in cognitive psychology have used various features to assist in performing the following visual tasks:

- *target detection*, where users attempt to rapidly and accurately detect the presence or absence of a "target" element that uses a unique visual feature within a field of distractor elements (Figure 1)

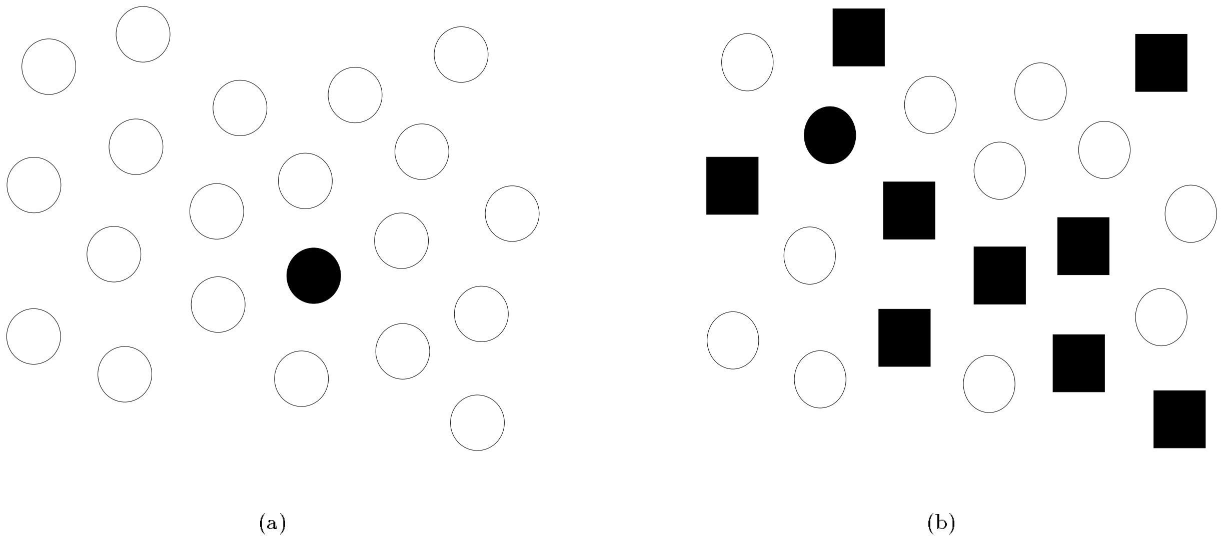


Figure 1: Examples of two target detection tasks: (a) target can be detected preattentively because it has a unique feature “filled”; (b) target cannot be detected preattentively because it has no visual feature unique from its distractors

- *boundary detection*, where users attempt to rapidly and accurately detect a texture boundary between two groups of elements, where all the elements in each group have a common visual feature (Figure 2)
- *counting and estimation*, where users attempt to count or estimate the number or percentage of elements in a display that have a unique visual feature

For our purposes, we will consider tasks which can be performed in less than 250 msec to be preattentive. Within this time frame an eye movement cannot be made to a new spatial location. This means preattentive tasks require only a “single glance” at the image being displayed. Non-preattentive tasks force the user to search serially through the display, examining each element in turn (*e.g.* conjunction target search). This means search time is proportional to the number of elements in the display, and can be easily made to violate the 250 msec bound if the number of elements is large enough. Conjunction search is one form of interference which may occur when we perform visualization.

A second type of interference has been studied by Callaghan and others. The visual system seems to prioritize features in order of importance. This means that the presence of visually “important” features can interfere with tasks which use lower priority features. For example, in Figure 2a, the vertical boundary defined by hue is detected preattentively, even though the shape of each element is random. However, in Figure 2b, it is difficult to detect the horizontal boundary defined by form. This is because hue varies randomly from element to element. If hue were fixed to a constant value for each element, the form boundary could be detected preattentively. Callaghan explains this phenomena by suggesting that the visual system assigns a higher importance to hue than to form [Callaghan, 1989; Callaghan, 1990] during boundary detection. Thus, a random hue interferes with form boundary detection, but a random form has no effect on hue boundary detection. A similar asymmetry exists between hue and intensity.

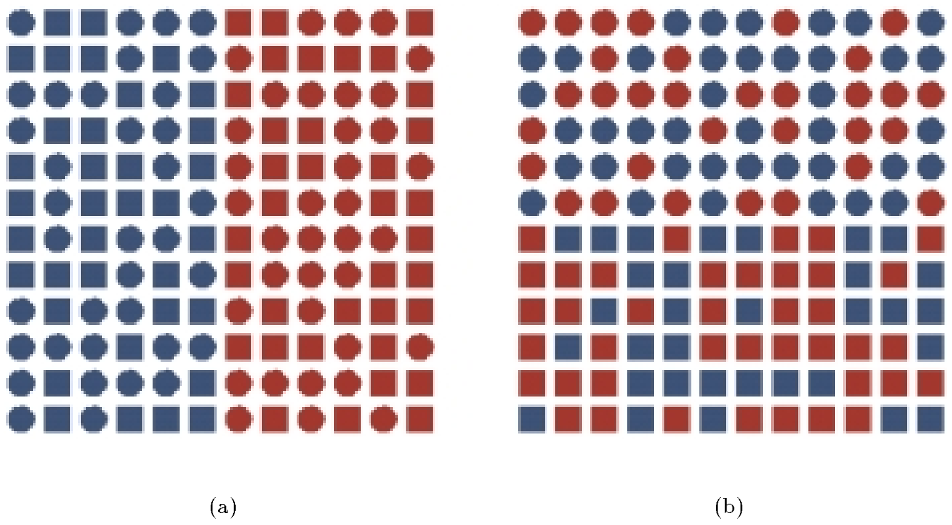


Figure 2: Region segregation by form and hue: (a) hue boundary is identified preattentively, even though form varies in the two regions; (b) random hue variations interfere with the identification of a region boundary based on form

Random hue has no effect on detecting boundaries defined by intensity. However, random intensity interferes with hue boundary detection. Callaghan concluded that intensity is more important than hue to the low-level visual system [Callaghan, 1984].

A final method of feature detection can be demonstrated through the use of emergent features. An emergent feature can be created by grouping several simpler shapes together. The emergent feature cannot be predicted by examining the simpler shapes in isolation (Figure 3). For example, in Figure 4a, the target element cannot be detected preattentively. However, by simply rotating one of the component elements, we create a new target with an emergent feature, non-closure, which is easily detected (Figure 4b).

Researchers continue to expand preattentive processing in a number of exciting directions. To date, most of the features used in preattentive tasks have been relatively simply properties (*e.g.* hue, orientation, line length, and size). Enns and Rensink have identified a class of three-dimensional elements that can be detected preattentively [Enns and Rensink, 1990; Enns, 1990]. They have shown that the three-dimensional orientation and direction of lighting is what makes the elements "pop-out" of the visual scene (Figures 3b and 3c). This is important, because it suggests that complex high-level concepts may be processed preattentively by the low-level visual system.

Another class of features being studied includes motion and depth. Initial research on motion and depth in preattentive processing was completed by Nakayama and Silverman [Nakayama and Silverman, 1986]. Their results showed that motion was preattentive. Moreover, stereoscopic depth could be used to overcome Triesman's conjunction effect. Like the work done by Enns and Rensink, this suggests that conceptually high-level information is being processed by the low-level visual system. Other work has focused on oscillating motion and its effect on conjunction search. [Driver et al., 1992] describe an experiment where subjects had to search for an X oscillating vertically

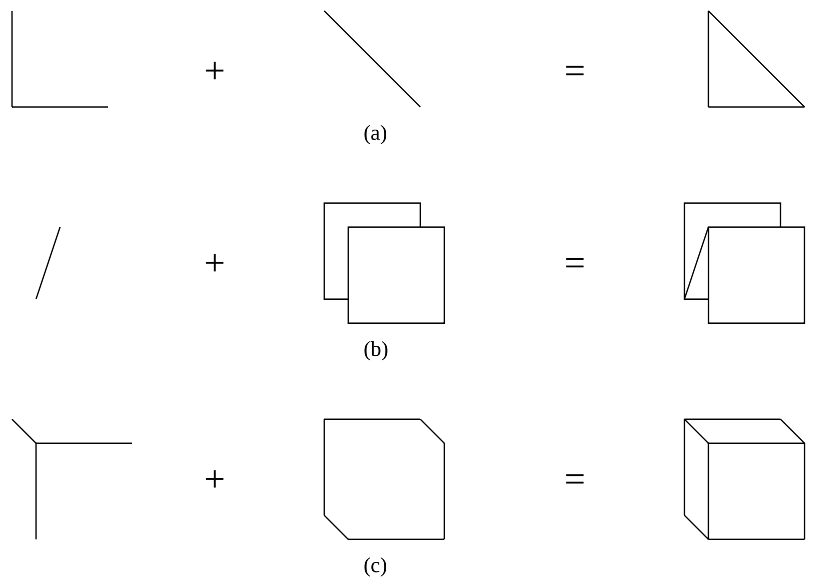
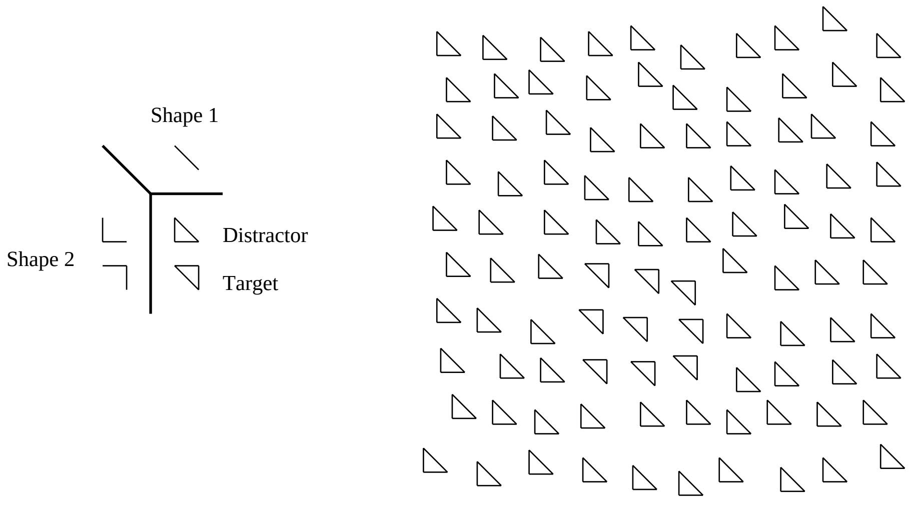


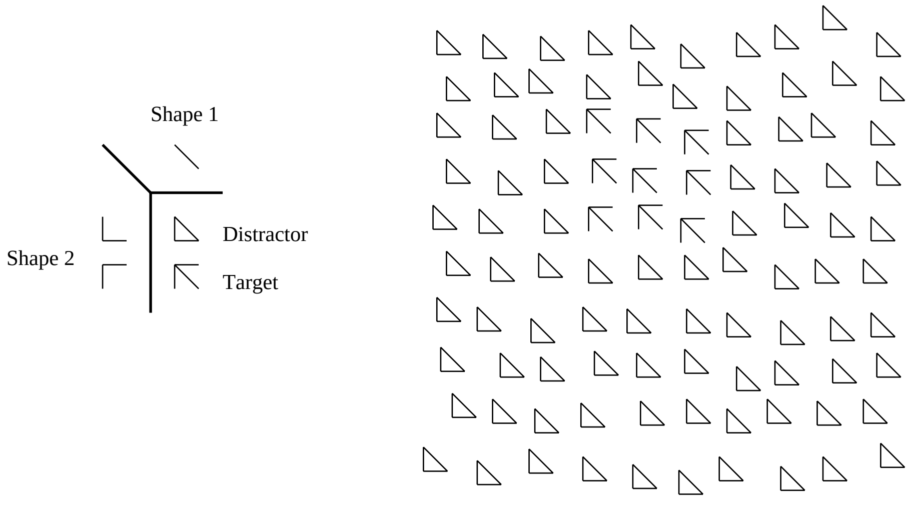
Figure 3: Combination of simple components to form emergent features: (a) closure, a simple closed figure is seen; (b) 3-dimensionality, the figure appears to have depth; (c) volume, a solid figure is seen

among O's oscillating vertically and X's oscillating horizontally. This task was preattentive if elements in the group oscillated coherently (*i.e.* vertically oscillating stimuli moved up and down together, horizontally oscillating stimuli moved left and right together). When elements oscillated "out of phase" with one another, subjects reverted to serial search. Finally, Braddick and Holliday showed that more complicated types of motion, such as divergence and deformation, required serial searching [Braddick and Holliday, 1987]. This implies that although motion itself is a preattentive feature, deformation and divergence involve targets with a number of different motions, each of which is shared by the distractors. Thus, a conjunction occurs and serial search is required to detect the presence or absence of a target.

In addition to new preattentive features, new tasks which can be performed preattentively have been investigated. For example, some research has been done on counting and estimation in preattentive processing. Varey describes experiments in which subjects were asked to estimate the relative frequency of white or black dots [Varey et al., 1990]. Her results showed that subjects could estimate in four different ways: "percentage" of white dots, "percentage" of black dots, "ratio" of black dots to white dots, and "difference" between the number of black and white dots. She also found that subjects consistently overestimated small proportions and underestimated large proportions. Estimation of relative frequency using hue and orientation was shown to be preattentive in experiments conducted in our laboratory [Healey et al., 1993; Healey et al., 1994]. Moreover, our results showed that there was no feature interaction. Random orientation did not interfere with estimation of targets with a unique hue, and random hue did not interfere with estimation of targets with a unique orientation. This is important because it suggests that hue and orientation can be used to encode two independent data values in a single display without causing interference.



(a)



(b)

Figure 4: A proper choice of initial components will form a target with an emergent feature which can be detected preattentively: (a) the target contains no unique emergent feature, so detecting the target group is difficult; (b) the target contains a unique emergent feature, non-closure, so the target group is easily detected

A number of scientists have proposed competing theories to explain how preattentive processing occurs, in particular Triesman's feature integration theory [Triesman, 1985], Julész' texton theory [Julész and Bergen, 1983], Quinlan and Humphreys' similarity theory [Quinlan and Humphreys, 1987], and Wolfe's guided search theory [Wolfe, 1994]. Our interest is in the use of visual features that have already been shown to be preattentive. We examined two such features, hue and form, and investigated their use for two common visualization tasks, boundary and target detection in a dynamic sequence of data frames.

4. EXPERIMENT 1: TEMPORAL BOUNDARY DETECTION

Through experimentation we sought to determine whether or not research in preattentive processing can help design more useful and intuitive scientific visualization tools. Specifically, we investigated whether preattentive tasks and interference effects extend to a real-time visualization environment, where frames of data are displayed one after another. Our first experiment addressed two general questions about preattentive features and their use in our visualization tools.

- *Question 1:* Is it possible for subjects to detect a data frame with a horizontal boundary within a sequence of random frames? If so, what features allow this and under what conditions?
- *Question 2:* Do Callaghan's feature preference effects apply to our real-time visualization environment? Specifically, does random hue interfere with form boundary detection within a sequence of frames? Does random form interfere with hue boundary detection within a sequence of frames?

These questions were designed to address the requirements described in Section 2. Detection of boundaries and groups is one example of a common data analysis task. If preattentive features can be used to help perform this task, VIS and other real-time applications could employ this technique for effective real-time visualization. Evidence that boundary detection and corresponding interference effects occur as expected in a real-time environment would imply that other preattentive tasks (*e.g.* target detection, counting, and estimation) might also extend naturally. The ability to encode multiple unrelated data values in a single display would allow users to visualize multidimensional datasets, or to "share" the display, but only in cases where no interference occurs.

We decided to examine two preattentive features, hue and form. This was done by running experiments which displayed 14×14 arrays of coloured circles and squares (Figures 6 and 7). These features are commonly used in existing visualization software. Both hue and form have been shown to be preattentive by Triesman, Julész, and others [Julész and Bergen, 1983; Triesman, 1985]. Moreover, Callaghan's research has shown that hue exhibits a strong interference effect over form during certain preattentive tasks. Understanding how hue and form interact in a preattentive visualization environment is important.

Two different hues were chosen from the Munsell colour space. The Munsell colour space was originally proposed

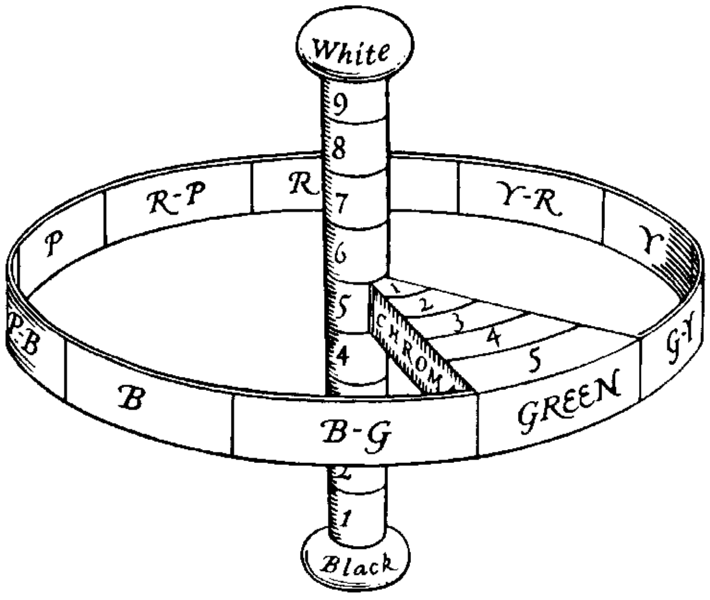


Figure 5: Munsell colour space, showing its three dimensions hue, value, and chroma (from *Munsell: A Grammar of Color*, New York, New York: Van Nostrand Reinhold Company, 1969)

by Albert H. Munsell in 1898 [Birren, 1969]. It was later revised by the Optical Society of America in 1943 to more closely approximate Munsell's desire for a functional and perceptually balanced colour system. A colour from the Munsell colour space is specified using the three "dimensions" hue, value, and chroma (Figure 5). Hue refers to ten uniquely identifiable colours such as red, blue, or blue-green. Individual hues are further subdivided into ten subsections. The number before the hue specifies its subsection (*e.g.* 5R, 2B, or 9BG). Value refers to a colour's lightness or darkness. It ranges from one (black) to nine (white). Chroma defines a colour's strength or weakness. Greys are colours with a chroma of zero. A chroma's range depends on the hue and value being used. A Munsell colour is specified by "hue value/chroma". For example, 5R 6/6 would be a relatively strong red, while 5BG 9/2 would be a weak cyan.

Since Munsell is a perceptually balanced colour space, it can be used to choose hues which are isoluminent. This is necessary, since intensity itself is a preattentive feature, and therefore must be equal for both hues. The exact hues we used were a red (Munsell 5R 7/8) and a blue (Munsell 5PB 7/8). Previous experiments ensured that the perceived difference between these two hues was large enough to be preattentively detected. Healey [1993] describes how this was done.

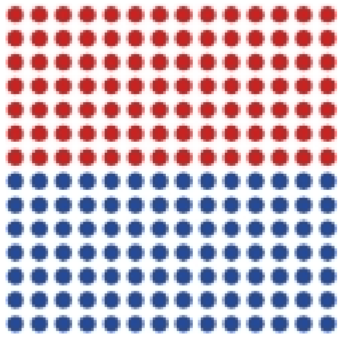
The experiment was split into two subsections B_1 and B_2 of 200 trials each. The first subsection tested a subject's ability to detect a horizontal boundary defined by hue (*i.e.* red and blue). The second subsection tested a subject's ability to detect a horizontal boundary defined by form (*i.e.* circle and square). Each trial was meant to simulate searching for a horizontal boundary while visualizing real-time data. A trial consisted of 18 separate data frames displayed to the subject one after another. Each frame was shown for a fixed amount of time (between 50 and 150 msec) which was chosen before the trial started. After viewing a trial, users were asked to indicate whether a frame containing a horizontal boundary had been present or absent. For "boundary present" trials, one of the 18 data frames was randomly chosen to contain a horizontal boundary. The remaining frames displayed a random pattern of features (with no horizontal boundary present). In "boundary absent" trials, all 18 frames displayed a random pattern of features; no frame contained a horizontal boundary.

Trials in each subsection were divided equally between control trials, where a secondary feature was fixed to a specific constant value, and experimental trials, where a secondary feature varied randomly from element to element. This allowed us to test for feature interference. Better performance in control trials versus experimental trials would suggest that using a secondary feature to encode an "irrelevant" data value interfered with a subject's boundary detection ability. We tested for both form interfering with hue boundary detection and hue interfering with form boundary detection. This experiment design gave us the following six subsections:

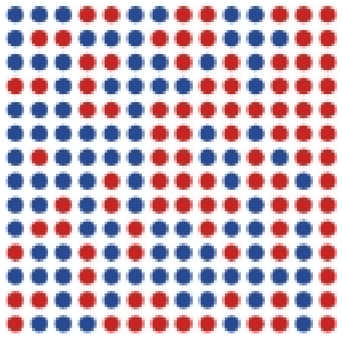
1. *hue-circle control*, horizontal boundary defined by hue, all elements are circles (Figures 6a- 6b).
2. *hue-square control*, horizontal boundary defined by hue, all elements are squares (Figures 6c- 6d).
3. *hue-form experimental*, horizontal boundary defined by hue, half the elements are randomly chosen to be circles, half to be squares (Figures 6e- 6f).
4. *form-red control*, horizontal boundary defined by form, all elements are red (Figures 7a- 6b).
5. *form-blue control*, horizontal boundary defined by form, all elements are blue (Figures 7c- 6d).
6. *form-hue experimental*, horizontal boundary defined by form, half the elements are randomly chosen to be red, half to be blue (Figures 7e- 6f).

Six subjects (five males and one female, aged 21 to 33) with normal or corrected acuity and normal colour vision volunteered to be tested. The experiments were conducted in the Computer Science Department's computer graphics laboratory, using a Silicon Graphics workstation equipped with a 21-inch colour display. The software used to conduct the experiments was written specifically to investigate preattentive visualization techniques. It used the display's vertical refresh to ensure accurate millisecond timing. Each subject completed both subsections of the experiment with three different frame exposure durations: 50 msec, 100 msec, and 150 msec.

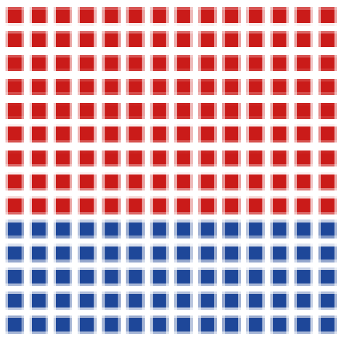
At the beginning of the experiment, subjects were shown a sample display frame. The experiment procedure and task were explained. Subjects were also shown how to enter their answers (either "present" or "absent") using the



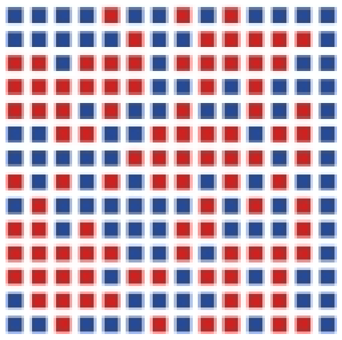
(a)



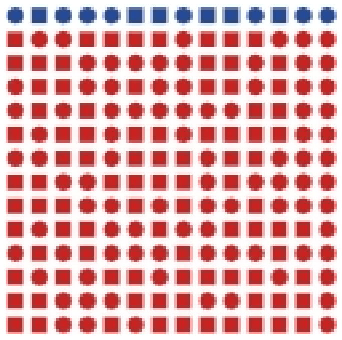
(b)



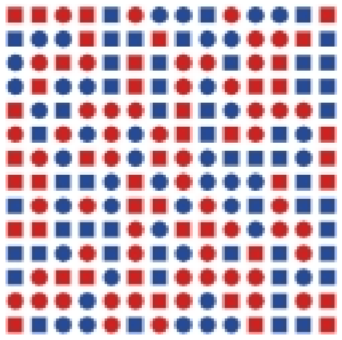
(c)



(d)

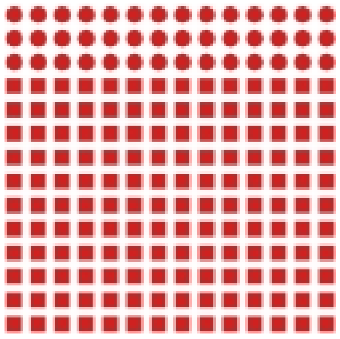


(e)

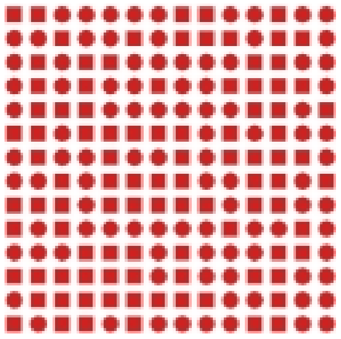


(f)

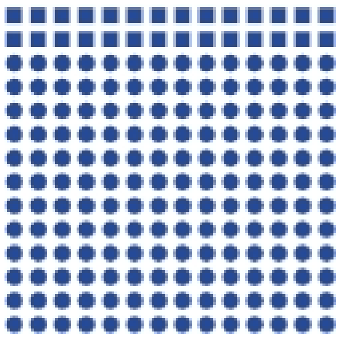
Figure 6: Example data frames from subsection B_1 of the boundary detection experiment (hue boundary detection): (a) control trial with all circles, boundary present; (b) control trial with all circles, boundary absent; (c) control trial with all squares, boundary present; (d) control trial with all squares, boundary absent; (e) experimental trial with random form, boundary present; (f) experimental trial with random form, boundary absent



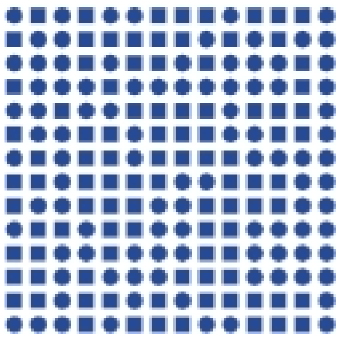
(a)



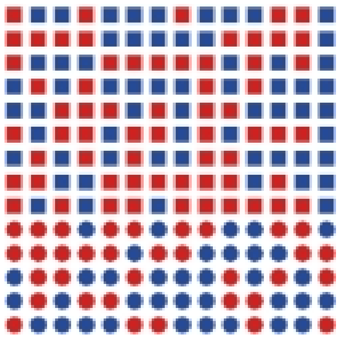
(b)



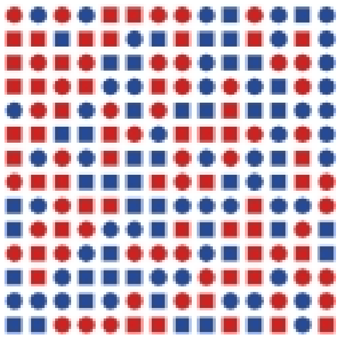
(c)



(d)



(e)



(f)

Figure 7: Example data frames from subsection B_2 of the boundary detection experiment (form boundary detection): (a) control trial with all red, boundary present; (b) control trial with all red, boundary absent; (c) control trial with all blue, boundary present; (d) control trial with all blue, boundary absent; (e) experimental trial with random hue, boundary present; (f) experimental trial with random hue, boundary absent

keyboard. Subjects began both subsections of the experiment with a set of practice trials. This consisted of 40 trials, 20 control trials split evenly between the two types of controls (*i.e.* ten trials with all circles and ten trials with all squares for subsection B_1, ten trials with all red and ten trials with all blue for subsection B_2) and 20 experimental trials. Ten control trials and ten experimental trials contained a horizontal boundary; the remaining trials did not. Exposure duration for practice trials was 100 msec per frame. The practice trials were designed give the subjects an idea of the speed of the trials and the experiment. Trials were displayed one after another, and subjects were asked whether a horizontal boundary had been present or absent after each trial. If a subject responded correctly, a plus sign was shown following the response. If a subject responded incorrectly, a minus sign was shown. Feedback (plus or minus) was displayed in the center of the screen for 400 msec, at a size of approximately twice that of a single data element (1.2 cm or subtending a visual angle of $1.1°$ at 60 cm).

Next, subjects completed the two experiment subsections B_1 and B_2. Each subsection consisted of 100 control trials and 100 experimental trials. Fifty control trials and 50 experimental trials contained a horizontal boundary; the remaining trials did not. The 200 trials from each subsection were presented to the subjects in a random order. Subjects were provided with an opportunity to rest after every 50 trials. Feedback (plus or minus) was displayed after every subject response. Subjects completed both subsections three times using three different exposure durations: 50 msec per frame, 100 msec per frame, and 150 msec per frame.

4.1 Results

The primary dependent variable examined was percentage error. Error was zero for trials where subjects responded correctly, and one for trials where they responded incorrectly. We began our analysis by dividing trials across the following experimental conditions, averaging response errors for each subject, then computing a mixed-factors ANOVA on the results:

- feature type; *hue* if a difference in hue defined the horizontal boundary, *form* if a difference in form defined the horizontal boundary
- trial type; *control* if the secondary feature was fixed to a constant value, *experimental* if it varied randomly from element to element
- block; BK_1 if a trial came from the first 100 trials the subject completed, BK_2 if it came from the last 100 trials
- exposure; *50*, *100*, or *150* msec, depending on a trial's display duration
- location of boundary frame (for experimental trials which contained a boundary frame); *front* if the boundary frame appeared during the first nine frames shown the the subject, *back* if it appeared during the last nine frames

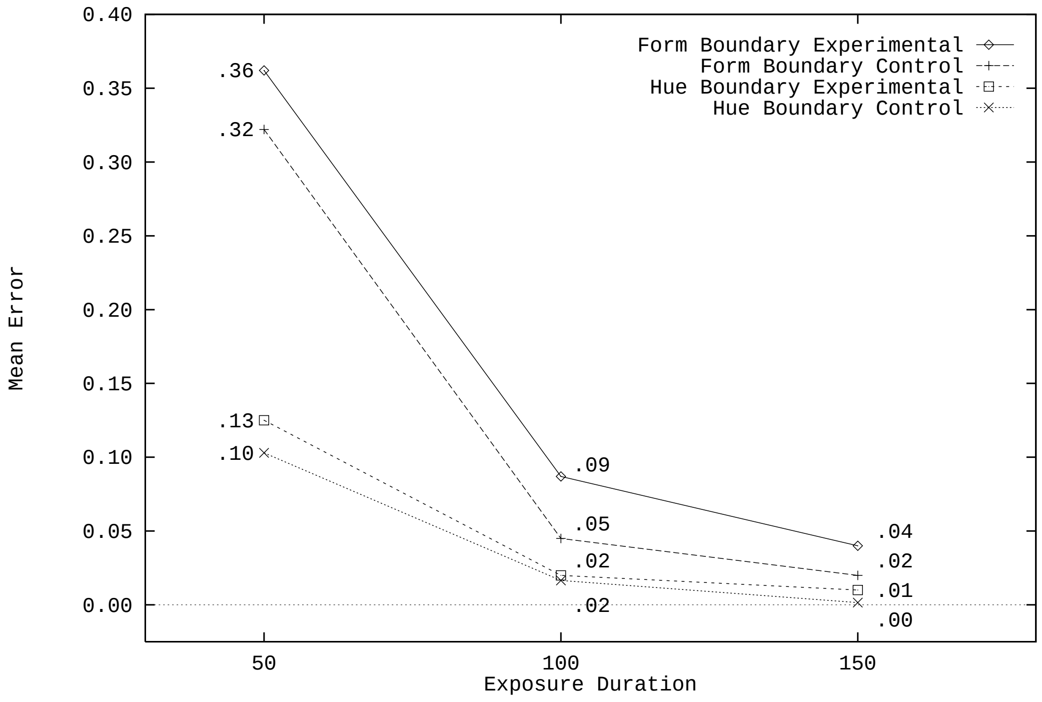


Figure 8: Graph of percentage error as a function of exposure duration for hue and form boundary trials; numbers represent exact percentage error values for each data point

Main effects with a p-value of less than 5% were considered significant. Results from the ANOVA suggested the following conclusions:

- rapid and accurate boundary detection can be performed using either hue or form; errors increased when exposure duration fell below 100 msec for both hue and form boundary detection
- form did not interfere with a subject's ability to detect a hue boundary at either 150 or 100 msec
- hue interfered with a subject's ability to detect a form boundary at both 150 and 100 msec
- accuracy was greater for hue than form, with this difference growing as exposure duration decreased
- there was no preference for the frame location of the target during either hue or form boundary detection

Figure 8 shows combined subject data for subsections B_1 (hue boundary detection) and B_2 (form boundary detection). The results indicate that hue boundary detection was quite accurate at all three exposure durations, with the most errors (about 13%) occurring in the experimental trials at 50 msec. Although errors for form boundary detection were uniformly higher than for hue boundary detection, subjects were still quite accurate at 100 msec (approximately 9%). Past that point, error rapidly approaches the chance limit of 50%, with an error rate of 36% at 50 msec.

Errors were generally higher for the form task than for the hue task, with $F(1,10) = 46.51$, $p = 0.001$. The feature type by exposure duration interaction of $F(2,20) = 45.54$, $p = 0.001$ was also significant. Additional F-values were computed to see how error varied across feature type (*i.e* hue and form) during the three exposure durations. In two of the three individual comparisons hue accuracy was significantly greater than form accuracy (p-values ranged from 0.02 to 0.001). The one exception concerned 150msec trials, where $F(1,5) = 2.04$, $p = 0.19$. Differences in accuracy increased as exposure duration decreased, suggesting that the perceived difference between our two hues was larger than the perceived difference between a circle and a square.

ANOVA results of $F(1,10) = 16.34$, $p = 0.02$ showed a significant difference in errors between control and experimental trials. A feature type by trial type interaction of $F(1,10) = 3.56$, $p = 0.09$ suggested interference was present during both hue and form boundary detection. Moreover, a trial type by exposure duration interaction of $F(1,10) = 1.10$, $p = 0.35$ indicated interference at all three exposure durations. Simple t-tests comparing control and experimental trials across exposure duration showed weak interference (at a significance level of 10%) during form boundary detection for 100 and 150 msec trials. Thus, the hue-on-form interference effect must be considered small, albeit consistent. Corresponding results for hue boundary detection found weak interference (at a significance level of 10%) for 50 msec trials. This is similar to Callaghan's [1989, 1990] static boundary detection experiments, although weak hue interference during form boundary detection was not reported in her results.

There was a significant exposure duration effect, $F(1,10) = 197.66$, $p = 0.001$. Individual F-values for the four conditions shown in Figure 8 (form boundary experimental, form boundary control, hue boundary experimental, and hue boundary control) were all $p = 0.001$. Fisher's protected least significant difference (PLSD) values identified significant differences between exposure durations $(50, 100)$ and $(50, 15)$ in all four conditions, but not between $(100, 150)$. We concluded that the high F-values were due to relatively higher errors during the 50 msec trials.

Finally, the results showed no boundary frame location preference for either hue ($F(1,5) = 2.37$, $p = 0.18$) or form ($F(1,5) = 0.05$, $p = 0.83$) boundary detection. Moreover, there was no consistent effect of trial block. Whereas errors actually increased from BK_1 to BK_2 in the hue condition, $F(1,5) = 17.44$, $p = 0.01$, they decreased (non-significantly) over time in the form condition, $F(1,5) = 5.51$, $p = 0.07$. There were no other significant interactions of the block factor with other factors of interest. We can draw no conclusions about the effects of practice or fatigue without performing additional experiments.

5. EXPERIMENT 2: TEMPORAL TARGET DETECTION

We continued our investigation of real-time preattentive visualization by studying temporal target detection. Our second experiment addressed two additional questions about preattentive features and their use in our visualization tools.

- *Question 1:* Is it possible for subjects to detect a data frame containing a unique target element in a sequence of random frames? If so, what features allow this and under what conditions?
- *Question 2:* Does any interference occur when viewing a dynamic sequence of data frames? Specifically, does random hue interfere with form target detection? Does random form interfere with hue target detection?

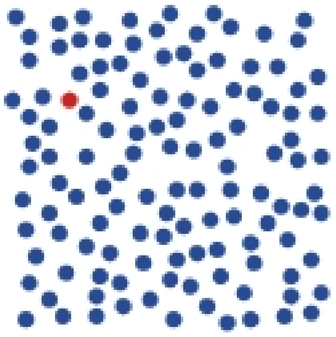
As with temporal boundary detection, these questions are specifically designed to address our visualization requirements. The ability to perform target detection using preattentive features would provide further justification for their use in VIS and other real-time applications. Our experiments also searched for any new types of interference which might occur as a result of viewing a dynamic sequence of data frames during visualization.

We chose to test the same two visual features (hue and form) used during the boundary detection experiments. Target detection experiments consisted of frames containing 125 elements (Figures 9 and 10). The position of the elements was held constant in every frame. Hue and form were the same as in the boundary detection experiments, specifically a red (Munsell 5R 7/8) and a blue (Munsell 5PB 7/8) hue, a circle and a square form.

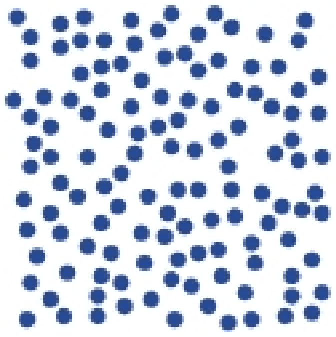
As in the first experiment, temporal target detection was split into two subsections T_1 and T_2 of 200 trials each. The first subsection tested a subject's ability to detect a target element defined by hue. The second subsection tested a subject's ability to detect a target element defined by form. Each trial was meant to simulate searching for a target element while visualizing real-time data. A trial consisted of 18 separate data frames, which were displayed to the subject one after another. Each frame was shown for a fixed amount of time (either 50 or 100 msec) which was chosen before the trial started. After viewing a trial, users were asked to indicate whether a frame containing the target element had been present or absent. For "target present" trials, one of the 18 data frames was randomly chosen to contain the target element. The remaining frames did not contain a target element. In "target absent" trials, none of the 18 frames contained a target element.

As with boundary detection, we tested for feature interference by dividing each subsection into control and experimental trials. In control trials, the secondary feature was fixed to a specific constant value; in experimental trials, it varied randomly from element to element. We tested for both form interfering with hue target detection, and hue interfering with form target detection. This gave us the following six subsections:

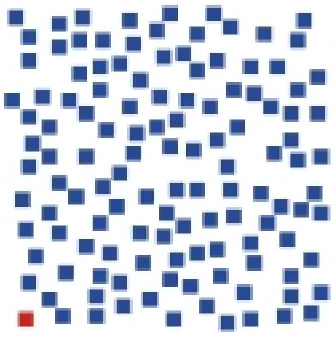
1. *hue-circle control*, target element is a red circle, all distractors are blue circles (Figures 9a- 9b).
2. *hue-square control*, target element is a red square, all distractors are blue squares (Figures 9c- 9d).
3. *hue-form experimental*, target element is a red circle, half the distractors are randomly chosen to be blue circles, half to be blue squares (Figures 9e- 9f).



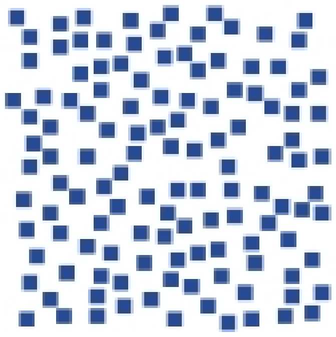
(a)



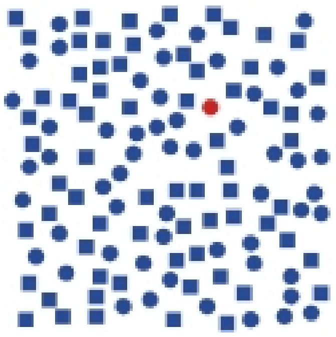
(b)



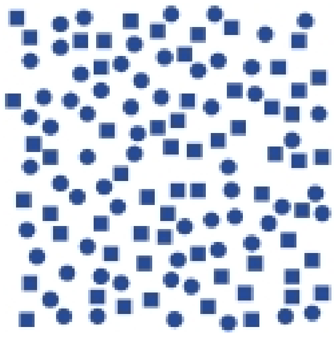
(c)



(d)



(e)



(f)

Figure 9: Example data frames from subsection T_1 of the target detection experiment (hue target detection): (a) control trial with all circles, target present; (b) control trial with all circles, target absent; (c) control trial with all squares, target present; (d) control trial with all squares, target absent; (e) experimental trial with random form, target present; (f) experimental trial with random form, target absent

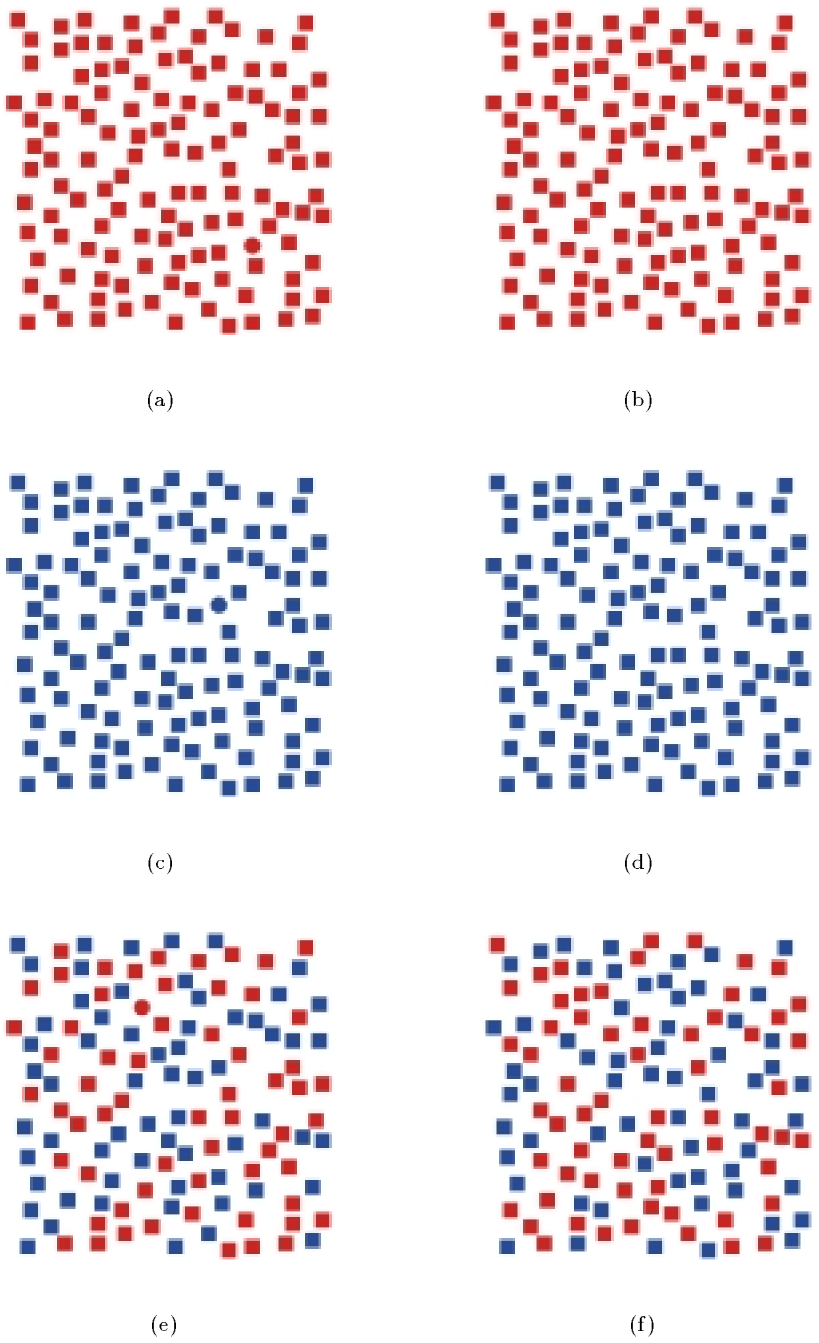


Figure 10: Example data frames from subsection T_2 of the target detection experiment (form target detection): (a) control trial with all red, target present; (b) control trial with all red, target absent; (c) control trial with all blue, target present; (d) control trial with all blue, target absent; (e) experimental trial with random hue, target present; (f) experimental trial with random hue, target absent

4. *form-red control*, target element is a red circle, all distractors are red squares (Figures 10a- 9b).
5. *form-blue control*, target element is a blue circle, all distractors are blue squares (Figures 10c- 9d).
6. *form-hue experimental*, target element is a red circle, half the distractors are randomly chosen to be red squares, half to be blue squares (Figures 10e- 9f).

Six subjects (five males and one female, aged 21 to 33) with normal or corrected acuity and normal colour vision were tested. Five of the six subjects also participated in Experiment 1. At the beginning of the experiment, subjects were shown a sample display frame. The experiment procedure and task were explained. Subjects were shown how to enter their answers (either present or absent) using the keyboard. Subjects began both subsections of the experiment with a set of practice trials similar to those for the boundary detection experiments. Exposure duration for practice trials was 100 msec per frame. Trials were displayed one after another. Subjects were asked whether a target element had been present or absent for each trial. Correct or incorrect responses were signaled by a plus or a minus sign.

Next, subjects completed the two experiment subsections T_1 and T_2. Each subsection consisted of 100 control trials and 100 experimental trials. Fifty control trials and 50 experimental trials contained a target element; the remaining trials did not. The 200 trials from each subsection were presented to the subjects in a random order. Subjects were provided with an opportunity to rest after every 50 trials. Feedback (plus or minus) was displayed after every response. Subjects completed both subsections two times using two different exposure durations: 100 msec per frame and 50 msec per frame.

5.1 Results

The primary dependent variable was again percentage error. A mixed-factors ANOVA was computed across the same conditions used for analysing boundary detection. The only difference was the number of possible values for exposure duration: *50* or *100* msec, depending on the trial's display duration. Results from the ANOVA can be summarized as follows:

- rapid and accurate target detection could be performed using hue at both 50 and 100 msec exposures
- similar rapid and accurate target detection based on form was possible only when hue was held constant
- form variations did not interfere with the ability to detect a hue-defined target
- hue variations did interfere with the ability to detect a form-defined target
- there was no preference for the frame location of the target during either hue or form target detection

Figure 11 graphs combined subject data for subsections T_1 (hue target detection) and T_2 (form target detection). The results show that hue target detection is very accurate at both exposure durations for control and experimental

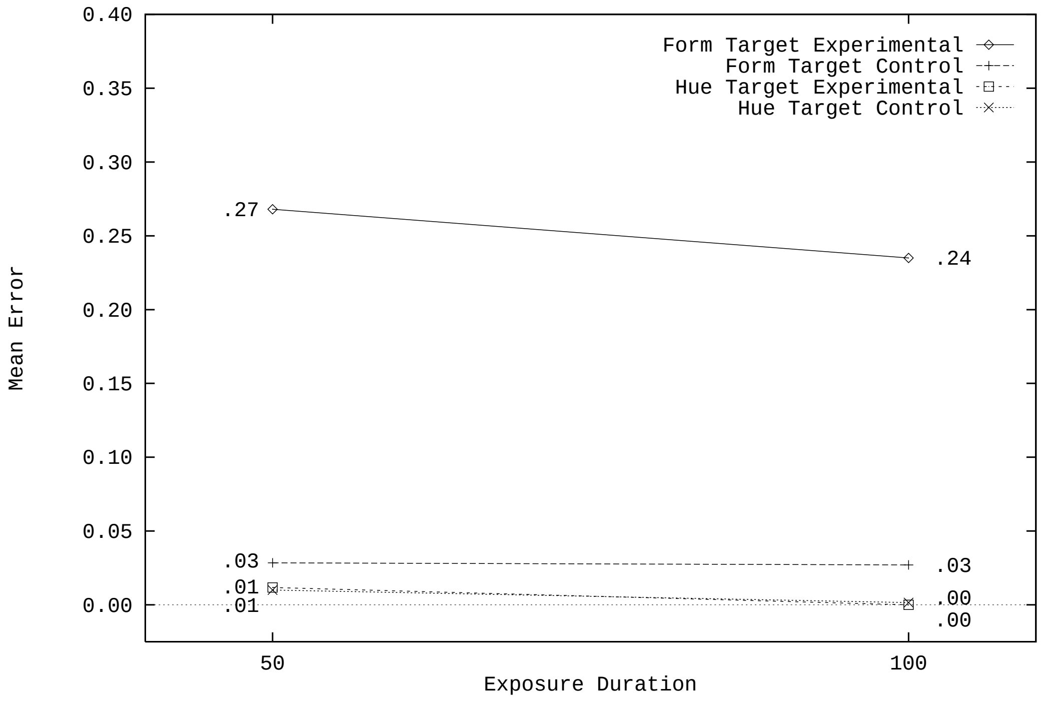


Figure 11: Graph of percentage error as a function of exposure duration for hue and form target trials; numbers represent exact percentage error values for each data point

trials (a maximum of 1% error). In contrast, form target detection was very accurate for control trials (3% error), but not for experimental trials (24% error at 100 msec and 27% error at 50 msec). There were no significant effects of exposure duration in any of the four conditions.

A feature type by trial type interaction of $F(1,10) = 62.52$, $p = 0.001$ suggests interference during one of the two target detection tasks. The difference in errors between control and experimental trials was significant in the form task, $F(1,5) = 103.58$, $p = 0.001$, but not in the hue task, $F(1,5) = 1.62$, $p = 0.26$. There was no feature type by trial type by exposure duration interaction, $F(1,10) = 0.39$, $p = 0.55$. Thus, as with our boundary detection task and Callaghan's [1989, 1990] static displays, random hue interferes with form target detection at both exposure durations, while random form does not interfere with hue target detection at either exposure duration. This provides further evidence for concluding that the perceived difference which can be obtained by using two hues is larger than the difference obtained from a circle and a square.

Figure 11 shows that errors were generally higher for the form task than for the hue task. ANOVA results supported this, with $F(1,10) = 46.51$, $p = 0.001$. There was no feature type by exposure duration interaction, $F(1,10) = 0.10$, $p = 0.76$. This means hue target detection was easier than form target detection at both exposure durations. Combined with hue interference during form target detection, this suggests that hue should be used as a primary feature when searching multidimensional data elements for a specific target.

As with the boundary detection task, there were no frame location effects for either hue, $F(1,5) = 1.26$, $p = 0.31$, or form, $F(1,5) = 1.04$, $p = 0.35$. The effects of trial block were again mixed. Errors were lower in BK_1 ($\overline{x} = 0.06$) than in BK_2 ($\overline{x} = 0.14$), $F(1,5) = 56.81$, $p = 0.001$ for form targets, but did not differ significantly for hue targets, $F(1,5) = 0.79$, $p = 0.41$. There were no significant interactions of trial block with other factors of interest.

6. PRACTICAL APPLICATIONS

As discussed in the introduction, our long-term goal is to provide general guidelines which can be used for the design of visualization systems. Behavioral experiments give us an opportunity to investigate the strengths and limitations of different designs in the context of simple datasets and visual tasks. Extending these results to a practical application is a non-trivial problem. However, even the limited results reported here can be used to help design real-world visualization applications. We discuss two such cases below. First, we describe a visualization system designed to help analyse results from salmon tracking simulations being conducted in the Department of Oceanography. Next, we discuss how our techniques might be used to visualize slice data from medical imaging systems like CT, PET, or MRI.

6.1 Salmon Tracking Simulations

Researchers in the Department of Oceanography at UBC are running various simulations to study the movement and migration patterns of sockeye salmon [Thomson et al., 1992; Thomson et al., 1994]. Salmon live in a number of different areas, including the western Canadian coast. A salmon's life-cycle consists of four stages. Salmon eggs hatch in freshwater lakes and rivers. The juveniles spend up to a year feeding and growing, after which they move downstream to the Pacific coast. They continue to an open ocean habitat, where they feed and grow for two to three years. Finally, the salmon begin their migration run. This consists of an open ocean stage back to the British Columbia coast and a coastal stage back to a freshwater stream to spawn. Salmon almost always spawn in the stream where they were born. Scientists now know that salmon find their stream of birth using smell when they reach the coast.

Many questions about how salmon move during their life-cycle remain unanswered. For example, scientists know that salmon which were born in a specific part of a river system feed in the open ocean as a group (or stock, as it is commonly called). Scientists want to identify the regions used by each stock of salmon, but these are currently undefined. Another question concerns the methods used to navigate from the open ocean back to the coast during the salmon's migration run. Oceanographer's simulations study the hypothesis that temperature, salinity, and ocean currents affect migration patterns. Our initial work in this area involved the design of tools to help analyse results from these simulations [Healey et al., 1993; Healey et al., 1994].

Another important area of investigation is the effect of ocean temperature on salmon movement patterns. Scientists know that salmon will not enter water which is above a certain temperature; in fact, they often sit directly along the "boundary temperature" gradient. The cutoff temperature changes in a seasonal manner. Questions the scientists would like to answer include: "What is the boundary temperature for a given season?", "How is the boundary temperature affected by changes in season?", and perhaps more fundamentally, "How do salmon measure water temperature?"

Datasets and simulation models to study these questions are currently being designed by the oceanographers. For example, simulated salmon can be "programmed" with a set of rules which dictate their movement. Movement rules might include how fast the fish swim, how much time they spend resting, and how environmental conditions like salinity and ocean current affect their swim patterns. Daily temperature values for uniform positions in the ocean from 1945 to 1994 are available in a large dataset. If a boundary temperature is chosen, positions in the ocean can be divided into two possible values, "hot" or "cold". The simulated salmon will then "swim" through the ocean, day by day, and oceanographers can check to see if the fish avoid hot areas as expected. The question we must answer is "How can we effectively visualize the simulation data, to allow the scientists to easily track the salmon as they move?"

Tracking groups of salmon involves both target detection (*i.e.* finding and following a group made up of multiple target elements) and boundary detection (*i.e.* identifying the boundary of the target group or groups). One possible visualization method is to encode salmon location and ocean temperature using hue and form. At each position in the ocean a fish is either present or absent, and the temperature is either hot or cold. Our experimental results suggest hue dominates form during both target and boundary detection. Since we are interested in following the position of the salmon, this data value should be encoded using hue. Ocean temperature, which is a secondary value, should be encoded using form.

Figure 12 shows four sample frames from our visualization tool. Ocean locations which contain salmon are coloured red. Empty locations are coloured blue. The group of salmon appears as a red "blob" moving through the ocean from frame to frame. Ocean temperature is encoding using form. Hot regions are drawn as circles, and cold regions are drawn as squares. It is interesting to note that this choice of data-feature mapping is somewhat counterintuitive. Normally, hot and cold are encoding using red and blue. In spite of this, it is very easy to follow the movement of the salmon group. For example, it is clear that the group begins to split in frames two and three. Closer inspection of these frames reveal that the fish are dividing to avoid a high temperature region.

Figure 13 shows the same four sample frames with the data-feature mapping reversed. Temperature is now encoded using hue, with red representing hot areas, and blue representing cold areas. Salmon position is drawn using form. Squares represent positions containing fish, and circles represent empty locations. In this example,

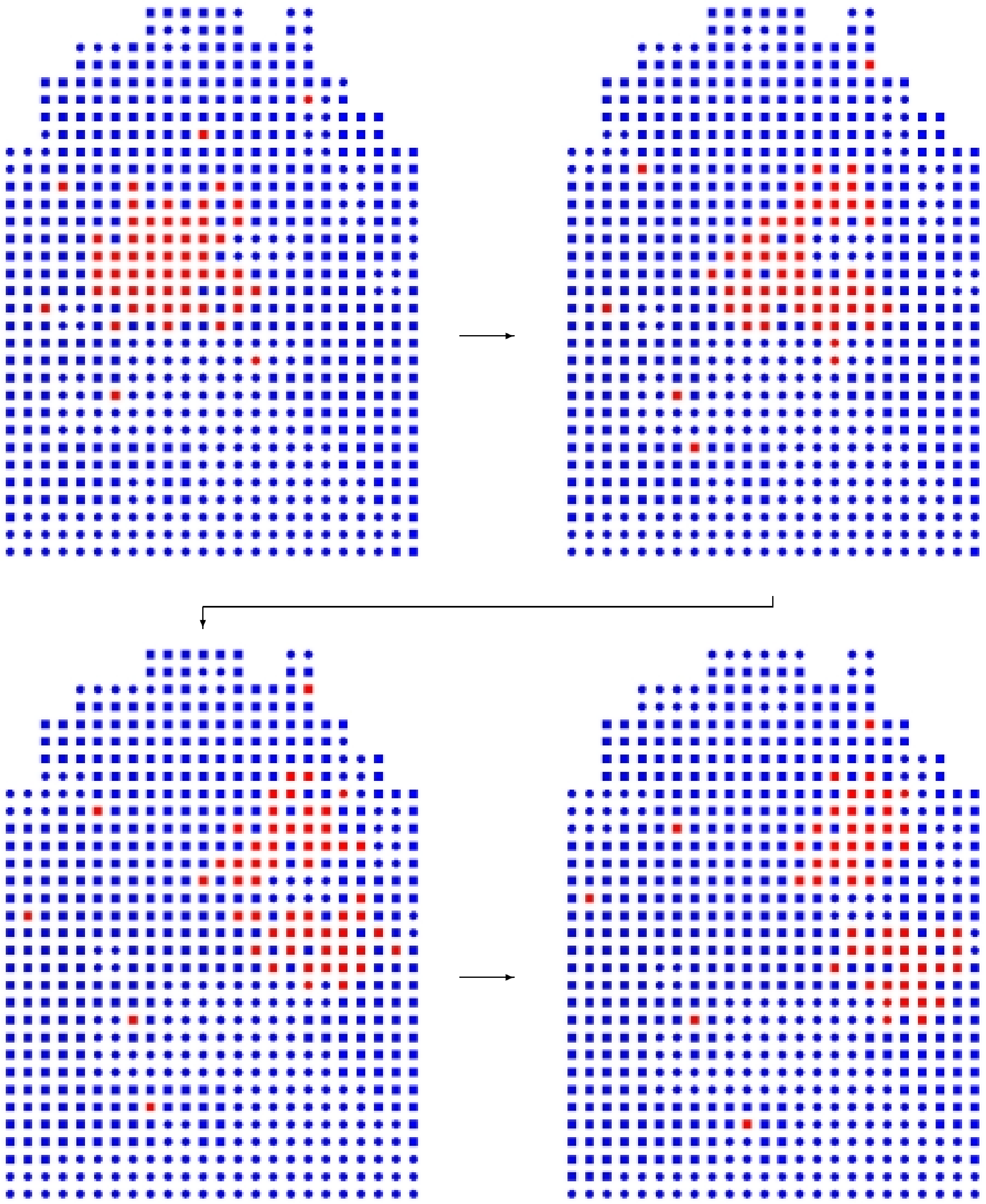


Figure 12: Sample frames from oceanography simulations: a group of salmon splits to avoid a high temperature region; salmon represented by hue (red for present, blue for absent) and temperature represented by form (circle for hot, square for cold)

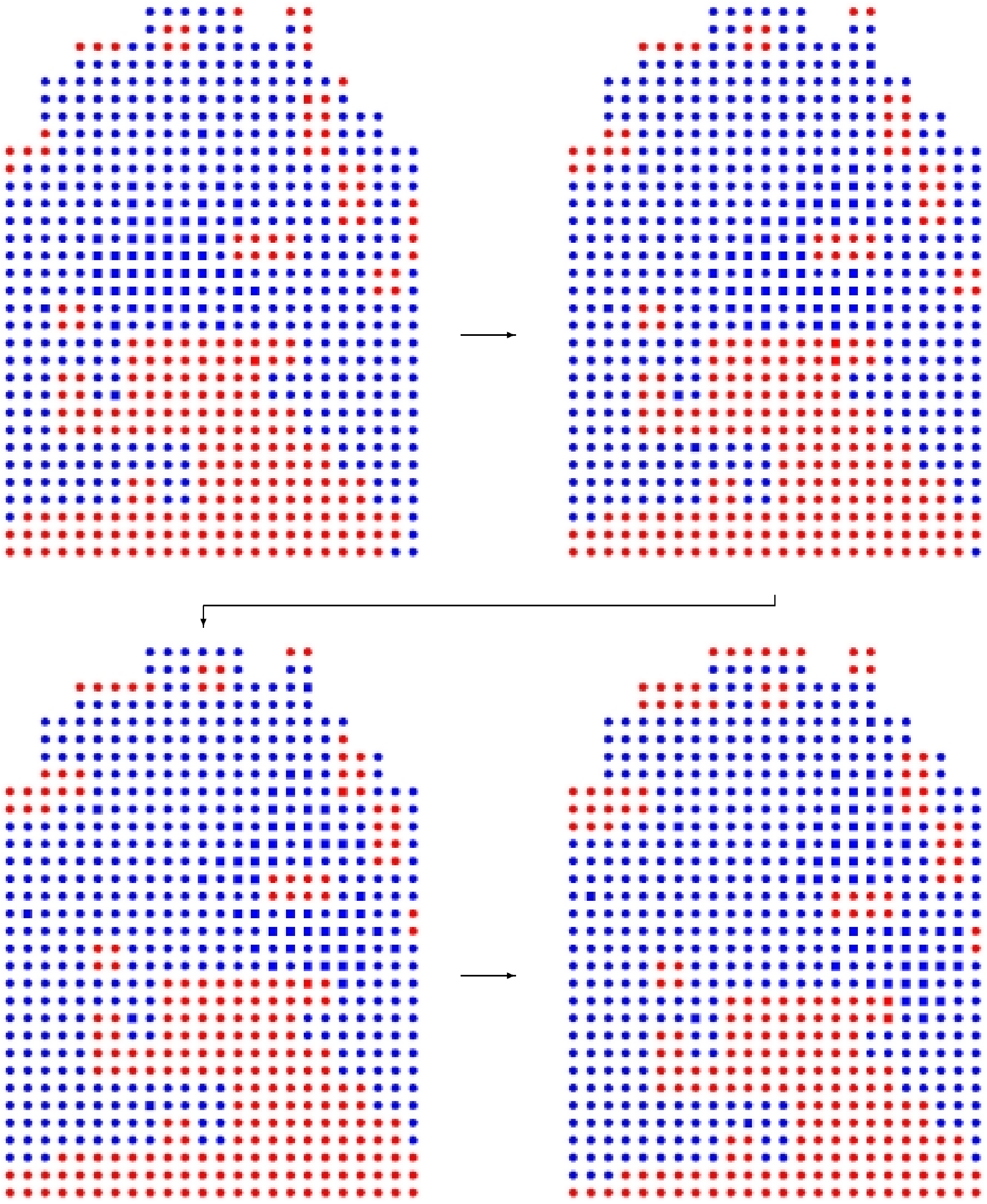


Figure 13: Sample frames from oceanography simulations: a group of salmon splits to avoid a high temperature region; salmon represented by form (square for present, circle for absent) and temperature represented by hue (red for hot, blue for cold)

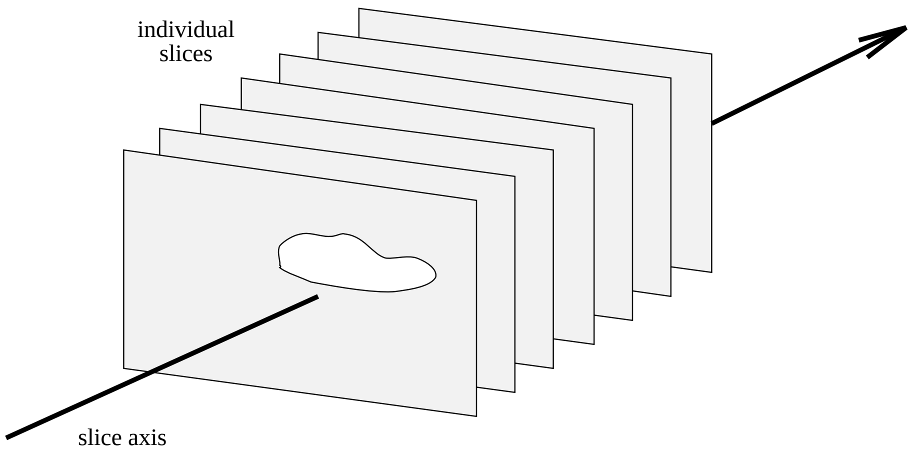


Figure 14: Example output from a medical imaging system, showing individual slices, the slice axis, and a region of interest marked on some or all of the individual slices

the prominent pattern is the ocean temperature, which moves with the current from frame to frame. It requires significantly more effort to find the salmon group. If frames were displayed at a reasonable speed (*e.g.* 100 msec per frame) it would be very difficult to follow the group as it moved, split, or merged.

6.2 Medical Imaging Systems

Advanced medical imaging systems like CT, MRI, and PET are now becoming reasonably wide-spread in hospitals and other medical institutions. A key concern is the effective analysis of output from these machines. Data is gathered by taking two-dimensional "slices" through the body. Orientation, interslice spacing, and other characteristics of the slices are controlled by the operator.

Various methods have been proposed to visualize slice data. One technique involves placing individual slices "side-by-side" for examination. Pairs or groups of slices are sometimes combined to study various hypotheses (*e.g.* the difference between pairs of PET slices are used to search for high activity regions in the brain).

Other methods attempt to reconstruct and display the original volume from the individual slices. Yagel et al. developed a method which interpolates slice data into a set of discretized volume elements (voxels) using a scan-conversion algorithm [Yagel et al., 1991]. The resulting volumes are rendered and displayed using a modified ray-tracing technique. Users can then perform a variety of constructive solid geometry operations (*e.g.* dissection, clipping, filtering, and thresholding) to see what is inside the object.

Another way of looking inside an object is to make the outer layers of the object semi-transparent. This allows us to see through these layers to whatever lies inside the model. Drebin et al. used this technique to visualize medical images. The body's outer surfaces (tissue and fat) are rendered as though they are transparent. The inner surfaces (bone) are solid [Drebin et al., 1988]. The resulting image is a skeleton surrounded by a semi-transparent "skin". Users can thus obtain visual information about both the surface and the interior of the body being displayed. Although Drebin divided his volumes into rectangular voxels, new methods have been proposed to simplify volume representation, thereby reducing the amount of time required to render and manipulate the volume [Ranjan and Fournier, 1994].

A simpler method of examining the volume is to move through it along some user-defined axis. At each step, a two-dimensional "cut" through the volume is displayed. If we choose to move along the slice axis, the problem is further simplified, since the original slices can simply be shown one after another. This is analogous to our dynamic visualization technique; each slice would be displayed for a fixed exposure duration, allowing the user to rapidly scan through the volume.

There are potential advantages to using this type of two-dimensional visualization method. It is often difficult with side-by-side displays to compare images which are not adjacent to one another. A dynamic visualization technique allows the user to detect subtle changes in the area, boundary, or makeup of regions of interest. Displaying two-dimensional frames requires very little computing power compared to reconstructing, displaying, and manipulating a three-dimensional volume. Two-dimensional displays avoid the problem of occlusion. Moreover, some tasks are easier to perform on two-dimensional images. Estimating area, for example, is easier than estimating volume.

Results from our experiments can be used to help choose a data-feature mapping for dynamically displaying the slice data. If users are trying to detect target elements, or find and track regions of interest, these should be encoded using hue. Secondary information can be displayed using different forms (*e.g.* circles and squares).

7. CONCLUSIONS AND FUTURE WORK

Results from our work provide a number of guidelines for the use of hue and form in real-time visualization. Hue can be used to perform rapid and accurate boundary and target detection. Form can be used to perform boundary detection, but it cannot be used as readily to perform target detection if a secondary data dimension is encoded with hue. If a user wants to perform real-time multidimensional visualization, hue should be used to encode the primary data dimension being investigated. Secondary data dimensions can be encoded with form. This will not interfere with boundary and target detection tasks performed using hue.

Multidimensional data often contains more than two data values to be encoded at each spatial location. As we have shown, tools which support the visualization of multiple data dimensions must deal with a potential interaction between some or all of the features being used to represent the dimensions. Rather than trying to avoid this, we can sometimes control the interaction and use it to our advantage. For example, Pickett and Grinstein have used texture to represent high-dimensional data; each dimension controls one aspect of a texture element displayed to the user. Another promising avenue of investigation involves emergent features. A careful choice of simple features will allow

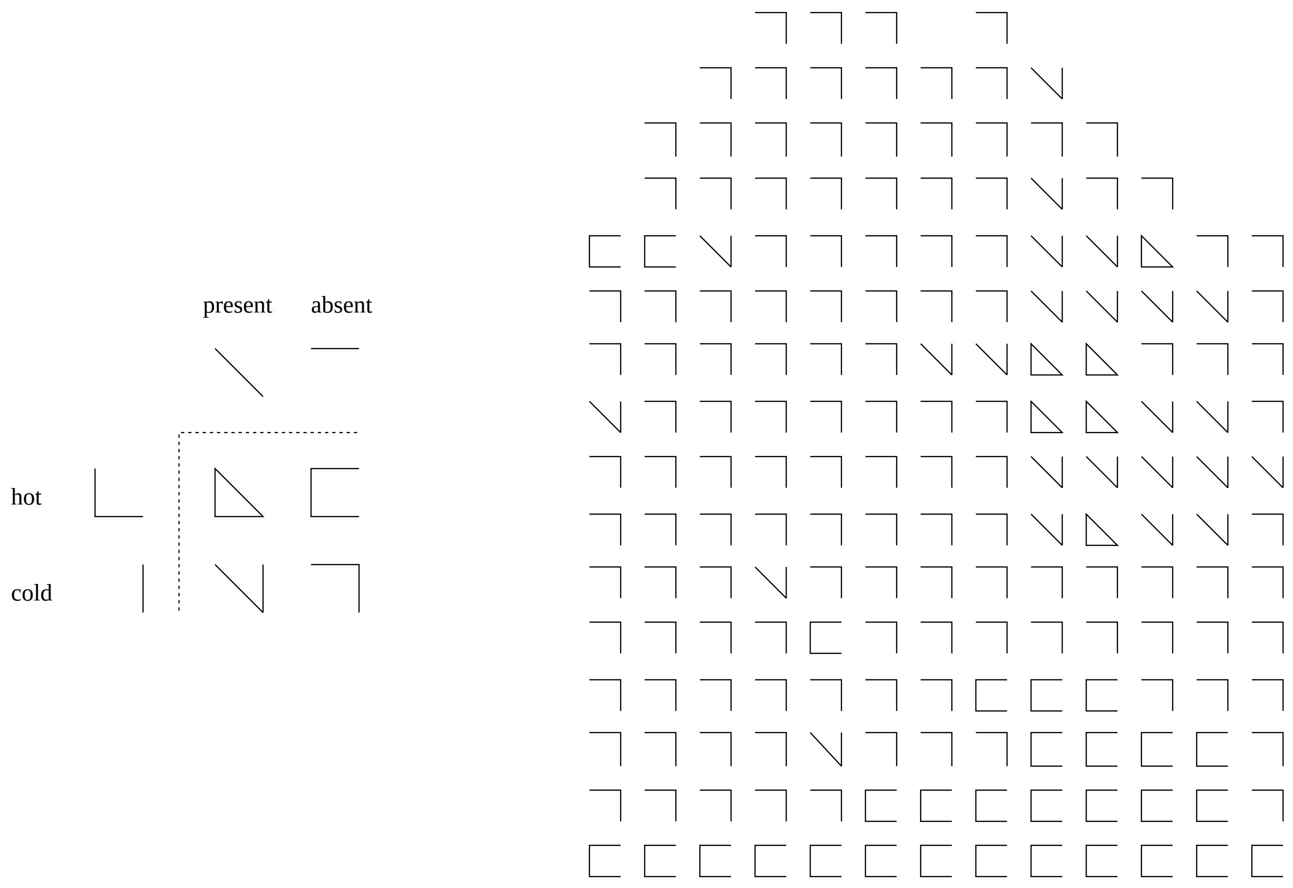


Figure 15: Example of output from salmon tracking simulations: salmon in hot ocean regions are displayed with an emergent feature closure; these can be detected since the three background objects do not use closure

a target element or a region of similar data elements to be detected preattentively [Pomerantz and Pristach, 1989], thereby signalling a correlation of variables in the data. Figure 15 shows one possible example of this technique. In the salmon tracking simulations discribed in section 6.1, scientists search for salmon entering hot ocean regions. This correlation of hot and present combines to form an an emergent feature, closure. Since background elements (hot-absent, cold-present, and cold-absent) do not have closure, the target salmon can be easily detected.

Another important question is how to encode dimensions which are more than two-valued (*i.e.* truly multi-valued or continuous dimensions). One potential method for representing a multi-valued dimension is to divide its visual feature space into more than two unique values. Research in preattentive processing has already studied some aspects of this problem. For example, it is easy to detect a tilted rectangle in a field of flat or upright rectangles (or vise-versa). However, in a situation where both the target and distractors are tilted (*e.g.* target is tilted 15° and distractors are tilted 75°), detection is much more difficult. Wolfe suggests orientation might be divisible into only four categories: steep, flat, left, and right [Wolfe et al., 1992]. Another example is the division of hue feature space. Experiments have studied the effect of dividing hue into multiple values. When a target used a hue which could be separated from the distractors by a straight line in colour space, detection was rapid and accurate. However, when the target used a hue which was collinear in this space with its distractors, detection was significantly more difficult [D'Zmura, 1991]. This suggests that hue feature space can be divided into multiple values, but that the target (or targets) must be "linearly separable" from their distractors. We plan to study both of these effects in the context of our visualization environment.

There are a number of important extensions which we would like to pursue, related specifically to real-time visualization using preattentive features. Additional preattentive tasks such as estimation and counting should be investigated to see if they extend to a temporally animated visualization environment. Other features, such as intensity and size, could also be tested to see if they can be used for the boundary and target detection tasks. Experiments using intensity and hue would confirm whether Callaghan's intensity-hue interference effects [Callaghan, 1984] are present in our real-time visualization tools.

ACKNOWLEDGMENTS

The authors wish to thank K.A. Thomson, W.J. Ingraham, and P.H. LeBlond of the Department of Oceanography, and M.C. Healey of the Westwater Research Centre for their advice and comments on the application of our work to their salmon migration simulations. The authors also wish to thank W.B. Cowan of the Computer Graphics Laboratory at the University of Waterloo for his advice on colour during the early stages of this work.

REFERENCES

Becker, R. A. and Cleveland, W. S. 1991. Take a broader view of scientific visualization. *Pixel 2*, 2, 42–44.

Bell, P. C. and O'Keefe, R. M. 1987. Visual interactive simulation—history, recent developments, and major issues. *Simulation 49*, 3, 109–116.

Bell, P. C. and O'Keefe, R. M. 1994. Visual interactive simulation: A methodological perspective. *Annals of Operations Research 53*, 321–342.

Birren, F. 1969. *Munsell: A Grammar of Color*. Van Nostrand Reinhold Company, New York, New York.

Braddick, O. J. and Holliday, I. E. 1987. Serial search for targets defined by divergence or deformation of optic flow. *Perception 20*, 345–354.

Callaghan, T. C. 1984. Dimensional interaction of hue and brightness in preattentive field segregation. *Perception & Psychophysics 36*, 1, 25–34.

Callaghan, T. C. 1989. Interference and domination in texture segregation: Hue, geometric form, and line orientation. *Perception & Psychophysics 46*, 4, 299–311.

Callaghan, T. C. 1990. Interference and dominance in texture segregation. In *Visual Search*, Brogan, D., Ed., 81–87. Taylor & Francis, New York, New York.

Clark, M. F. 1991. WITNESS: Unlocking the power of visual interactive simulation. *European Journal of Operations Research 54*, 293–298.

Collins, N. and Watson, C. M. 1993. Introduction to ARENA. In *Proceedings of the 1993 Winter Simulation Conference*, 205–212, Los Angeles, California.

Concannon, K. and Becker, P. 1990. A tutorial on GENETIK simulation and scheduling. In *Proceedings of the 1990 Winter Simulation Conference*, 140–144, New Orleans, Louisiana.

Drebin, R. A., Carpenter, L., and Hanrahan, P. 1988. Volume rendering. *Computer Graphics 22*, 4, 65–74.

Driver, J., McLeod, P., and Dienes, Z. 1992. Motion coherence and conjunction search: Implications for guided search theory. *Perception & Psychophysics 51*, 1, 79–85.

D'Zmura, M. 1991. Color in visual search. *Vision Research 31*, 6, 951–966.

Earle, N. J., Brunner, D. T., and Henriksen, J. O. 1990. PROOF: The general purpose animator. In *Proceedings of the 1990 Winter Simulation Conference*, 106–108, New Orleans, Louisiana.

Enns, J. T. 1990. The promise of finding effective geometric codes. In *Proceedings Visualization '90*, 389–390, San Francisco, California.

Enns, J. T. and Rensink, R. A. 1990. Sensitivity to three-dimensional orientation in visual search. *Psychology Science 1*, 5, 323–326.

Fiddy, E., Bright, J. D., and Hurrion, R. D. 1981. SEE-WHY: Interactive simulation on the screen. In *Proceedings of the Institution of Mechanical Engineers C293*, 167–172.

Fournier, A. 1994. From the look of things. In *Proceedings Graphics Interface '94*, 157–166.

Garrison, W. J. 1990. A brief SIMSCRIPT II.5 tutorial. In *Proceedings of the 1990 Winter Simulation Conference*, 115–117, New Orleans, Louisiana.

Healey, C. G., Booth, K. S., and Enns, J. T. 1993. Harnessing preattentive processes for multivariate data visualization. In *Proceedings Graphics Interface '93*, 107–117, Toronto, Canada.

Healey, C. G., Booth, K. S., and Enns, J. T. 1994. High-speed visual estimation using preattentive processing. *ACM Transactions on Computer-Human Interaction* (submitted).

Henriksen, J. O. 1993. SLX, the successor to GPSS/H. In *Proceedings of the 1993 Winter Simulation Conference*, 263–268, Los Angeles, California.

Hurrion, R. 1976. *The Design, Use, and Required Facilities of an Interactive Visual Computer Simulation Language to Explore Production Planning Problmes.* Ph.D. thesis, The University of London, England.

Hurrion, R. D. 1980. An interactive visual simulation system for industrial management. *European Journal of Operations Research 5*, 86–93.

Julész, B. and Bergen, J. R. 1983. Textons, the fundamental elements in preattentive vision and perception of textures. *The Bell System Technical Journal 62*, 6, 1619–1645.

Kalasky, D. R. 1991. Computer animation with CINEMA. In *Proceedings of the 1991 Winter Simulation Conference*, 122–127, Phoenix, Arizona.

Kaufman, A. and Hanani, M. Z. 1981. Converting a batch simulation program to an interactive program with graphics. *Simulation 36*, 4, 125–131.

Lilegdon, W. R. and Ehrlich, J. N. 1990. Introduction to SLAMSYSTEM. In *Proceedings of the 1990 Winter Simulation Conference*, 77–79, New Orleans, Louisiana.

McCormick, B. H., DeFanti, T. A., and Brown, M. D. 1987. Visualization in scientific computing—a synopsis. *IEEE Computer Graphics and Applications 7*, 7, 61–70.

Melamed, B. and Morris, R. J. T. 1985. Visual simulation: The performance analysis workstation. *IEEE Computer 18*, 87–94.

Murgiano, C. 1990. A tutorial on WITNESS. In *Proceedings of the 1990 Winter Simulation Conference*, 177–179, New Orleans, Louisiana.

Nakayama, K. and Silverman, G. H. 1986. Serial and parallel processing of visual feature conjunctions. *Nature 320*, 264–265.

O'Keefe, R. and Davies, R. 1986. A microcomputer system for simulation modeling. *European Journal of Operational Research 24*, 23–29.

O'Reilly, J. J. and Whitford, J. P. 1990. SLAM II tutorial. In *Proceedings of the 1990 Winter Simulation Conference*, 72–76, New Orleans, Louisiana.

Pomerantz, J. and Pristach, E. A. 1989. Emergent features, attention, and perceptual glue in visual form perception. *Journal of Experimental Psychology: Human Perception & Performance 15*, 4, 635–649.

Quinlan, P. T. and Humphreys, G. W. 1987. Visual search for targets defined by combinations of color, shape, and size: An examination of task constraints on feature and conjunction searches. *Perception & Psychophysics 41*, 5, 455–472.

Ranjan, V. and Fournier, A. 1994. Volume models for volumetric data. *IEEE Computer 27*, 7, 28–36.

Rosenblum, L. J. 1994. Research issues in scientific visualization. *IEEE Computer Graphics and Applications 14*, 2, 61–85.

Thomson, K. A., Ingraham, W. J., Healey, J. C., LeBlond, P. H., Groot, C., and Healey, C. G. 1992. The influence of ocean currents on the latitude of landfall and migration speed of sockeye salmon returning to the Fraser River. *Fisheries Oceanography 1*, 2, 163–179.

Thomson, K. A., Ingraham, W. J., Healey, J. C., LeBlond, P. H., Groot, C., and Healey, C. G. 1994. The influence of ocean currents on return timing of Fraser River sockeye salmon. *Canadian Journal of Fisheries and Aquatic Sciences* (in press).

Treinish, L. A., Foley, J. D., Campbell, W. J., Haber, R. B., and Gurwitz, R. F. 1988. Effective software systems for scientific data visualization. *Computer Graphics 23*, 5, 111–136.

Triesman, A. 1985. Preattentive processing in vision. *Computer Vision, Graphics and Image Processing 31*, 156–177.

Varey, C. A., Mellers, B. A., and Birnbaum, M. H. 1990. Judgments of proportions. *Journal of Experimental Psychology: Human Perception & Performance 16*, 3, 613–625.

Ware, C. and Beatty, J. C. 1988. Using colour dimensions to display data dimensions. *Human Factors 30*, 2, 127–142.

Wolfe, J. M. 1994. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin & Review 1*, 2, 202–238.

Wolfe, J. M. and Franzel, S. L. 1988. Binocularity and visual search. *Perception & Psychophysics 44*, 81–93.

Wolfe, J. M., Friedman-Hill, S. R., Steward, M. I., and O'Connel, K. M. 1992. The role of categorization in visual search for orientation. *Journal of Experimental Psychology: Human Perception & Performance 18*, 34–49.

Yagel, R. A., Kaufman, A., and Zhang, Q. 1991. Realistic volume imaging. In *Proceedings Visualization '91*, 226–231, San Diego, California.