Adding Colour to the Workstation Environment

William Cowan
Department of Computer Science
University of Waterloo
Waterloo, Ontario, N2L 3G1

ABSTRACT

Colour is an increasingly common feature of computer workstations. New ideas, algorithms and knowledge are needed to harmonize the user's desire for improved aesthetic appeal and the programmer's desire for an interface of greater functionality. This paper discusses several areas in which colour leads to novel functionality, visual search, grouping by colour and colour coding. The current obstacles to wider use of these effects are scientific: research is needed to develop colour specifications that allow their precise exploitation. As soon as they are overcome, however, conflicts between aesthetic and functional use of colour will become important. The solution of that problem will require a much better understanding of visual context. Visual context is especially important for window systems, the user interface of choice on modern workstations, since it is desirable that applications should be able to run unaware of other applications with which they share the display surface. If the window system can create a separate visual context for each application many colour problems are solved, but not all. Non-local properties of colour may still require information sharing among applications.

RESUME

La couleur est une caractéristique de plus en plus répandue des postes de travail informatiques. L'usager recherche l'esthétique, tandis que le programmeur s'efforce de créer un interface toujours plus fonctionnel; la reconciliation des deux nécessite de nouvelles idées, algorithmes et connaissances dans le domaine. Cet article explore quelques domaines ou la couleur parvient a innover la fonctionnalité, la recherche visuelle, le groupement par couleur et le codage par couleur. L'usagé plus répandu de ces effets fait présentement face a des obstacles d'ordre scientifique: cela réquiert le développement de specifications de coloris permettant leur exploitation exacte. Une fois surmontés, le conflit entre l'usage esthétique vs fonctionnel des couleurs prendra de l'importance. La résolution de ce problème exigera une compréhension bien plus poussée du contexte visuel. Le contexte visuel est particulièrement important en ce qui concerne les "systèmes de fenêtres", l'interface-usager de choix sur les postes de travail plus récents, puisqu'il est avantageux qu'une application puisse fonctionner indépendamment des autres applications avec lesquelles elle partage la surface d'affichage. Un nombre de problèmes associés avec la couleur sont résolus si le système de fenêtres peut fournir un contexte visuel distinct pour chaque application, mais pas tous. Certaines proprietés non-locales des couleurs peuvent encore nécessiter un partage d'information parmi les applications.

I. INTRODUCTION

A. Modern workstation hardware

One of the most interesting developments currently taking place in computing is the trend toward more distributed systems. These changes are the culmination of a process that began in the mid 1960s, a process that has moved computation away from a shared central facility towards an environment in which each worker is computationally self-sufficient. Graphics has undergone a similar evolution, from centralized plotting and display facilities to local display capabilities, at least for soft copy. The current state-of-the-art in this process is the engineering workstation, which combines local computation with high quality soft copy graphics. Workstations span a wide range with fuzzy limits, from fully configured personal computers to machines that bill themselves as supercomputers. What they all have in common is the integration of text and graphics on a bit-mapped display, a more or less graphical user interface and a display surface organized as windows. Most machines of this type have bilevel displays capable of only two values, white and black, at each pixel. But increasing numbers of workstations are available with colour displays. The purpose of this talk is to explore some of the problems and opportunities inherent in this new development, particularly those that occur in the graphical interface.

Four levels of colour are available in the workstation market. The bilevel displays mentioned above have one bit of graphics memory per pixel and are capable of two colours, usually black and white. Visual textures are commonly used on these machines to perform colour functions and are surprisingly similar. A few grey scale machines, having monochrome displays with more than two levels of grey, exist, usually with eight bits per pixel for 256 levels of grey. More common are pseudocolour machines which combine memory of this depth with a colour map to make available 256 discrete colours from a much larger palette. At the top of the range are full colour machines with 24 bits per pixel, making possible the independent selection of 256 levels of each of red, green and blue. These capabilities are most applicable to CRT-based, soft copy displays. In addition, colour printers are becoming more common, so much so that they are likely to be normal workstation adjuncts in the near future.

A capability important in workstations is the ability to run more than one application program at a time, usually in a separate window. This feature creates one of the most challenging problems in using colour, since the interaction between information displayed by different applications must be controlled. The window system has the responsibility for managing a visual environment in which each application can run independently of the other applications. Thus, much of the discussion to follow considers functional properties that any window management software should provide. Implementation details are left as an exercise.

B. Why do users want colour?

It is interesting to watch as users experiment with workstations that have colour. Usually, they begin by creating an uncontrolled riot of colour, working hard to obtain as much colour as possible in applications and in the window software. Why? The obvious interpretation is that colour enriches the visual environment provided by the workstation. In addition, the ability to create personal colour schemes allows the user to express his or her individuality in and through the display. These motivations are essentially identical to the ones that impel people to paint their living rooms or to hang pictures on the walls of their offices. Later on users seem to notice the deficiencies in their colour choices and then divide into two groups. Those willing to spend the time tune their colour choices to obtain a more aesthetic display personality. Other users reduce their colour usage until colour plays only a functional role.

In contrast, application creators seem to focus on using colour to provide additional information to the user, or to enhance existing information. When user control of colour schemes is made available, it is often done, as in X [1], through a level of indirection, like foreground, background, highlight, which assumes implicitly that the user wants colour to reinforce functional categories in the displayed information.

This divergence of interest appears to demonstrate an opposition between function and aesthetics. For example, if the user is allowed to colour window borders red how will the application's use of red as a warning colour possibly work? In this view the conflict lies between programmers and users and rejecting aesthetic colour for the benefit of functional colour is anti-user. I will suggest that the conflict can be resolved, based on a better understanding of functional colour. First, however, we must see that there are additional complications when a variety of applications share the display surface.

C. The user is a resource shared by applications

The key concept when several applications share a workstation is synchronization. Some system resources, like printers, can be used by only a single application at a time; otherwise output from several applications is inappropriately mixed together. System software spools the output, thereby serializing access to the device. The display cannot be treated like a printer since it is essential that several applications have access to it at the same time. The solution is to break the display into a large set of resources, one for each pixel. Sets of pixels are then parcelled out by window system software and made available for the output of applications. Since the system interjects itself into the graphics pipeline by managing the clipping region, applications are able to write to the screen independently of one another. The next two paragraphs describe several other synchronization problems. In each of them extra hardware turns out to be a possible solution. Such is not the case for pixels, since an important feature of a window system is centralization of diverse information in a single location.

The colour map is a more complicated resource to manage since it must be used simultaneously by all applications. A dynamic strategy similar to the pixel strategy is possible. The colour map is broken into a large set of small resources that are handed out each time an application needs another colour. Unfortunately, this strategy does not fail gracefully when the resource is exhausted. Thus, a static strategy is often better. The system decides in advance on a reasonable compromise among the demands it is likely to receive and sets up the colour map accordingly. It then presents an application with the colour closest to the one requested. The compromise allows all applications to share the colour map. Note that it is always possible to improve the performance of static strategies by providing extra hardware, additional colour maps in this case.

There is a final reason for synchronization, atomic access. If, for example, an application sets a write mask then writes through it, no other application may change the mask before the write is complete. Synchronization, provided by the system, serializes access to provide the atomic access needed. Here, as for the colour maps, extra hardware provides significant assistance.

It is important to notice that the user's perceptual systems are a shared resource, just like the workstation hardware, and that synchronization issues apply to them equally. Here are a few examples. (1) The human visual system parses a scene into a set of objects and extracts information from each object in turn. Information must remain consistent within each object during the time that information may be extracted from it. (2) The visual system is capable of associating meaning with displayed colours. Applications can agree on a static strategy for sharing a common colour/meaning association or colours can be dynamically allocated to applications by the system software. (3) The state of adaptation of the user's visual system is an important factor affecting what colour is perceived. It is controlled partly by the contents of the application window, partly by the remainder of the visual environment. An application can manipulate the display so as to create the adaptive state appropriate to an item of information. Clearly this adaptive manipulation and the ensuing display of information must be an atomic operation.

We choose to call such issues "synchronization of content" in contrast with synchronization of hardware devices. In window systems synchronization is generally used to describe the process by which the system controls access to the media, pixels, colour maps, and so on, used by applications for communication with the user. By contrast the form of synchronization at issue here relates to the content of information displayed in the windows. Ideally an application should be able to display output without regard for the contents of windows owned by other applications. Then it would be the responsibility of the system software to manage the visual interactions among all the windows so that the intent of each application was satisfied. Unfortunately, since the perception of colour is based on global properties of the stimulus, all pixels in the display interact. Thus, given currently available synchronization techniques it is possible to synchronize content only if applications communicate the information they wish displayed to the system software and allow it to take things from there. Insufficient understanding of perceptual information processing is currently an insurmountable obstacle to this objective.

D. A few important facts about colour

A better understanding of the effect of colour on multi-window display surfaces is possible in the context of a few basic facts about colour. These facts are based on the opponent channels model of colour vision [2]. The model includes two stages of processing. The first consists of the three cone types, red, green and blue. Each absorbs photons with a characteristic responsivity, creating the trivariate colour signal. In the second stage these three signals are combined into three opponent channels. One is achromatic, responsive to most wavelengths of light because it is excited by all three cone channels [3]. The other two channels are chromatic, red/green, which is excited by red cones and inhibited by green cones, and yellow/blue which is excited by red and green cones and inhibited by blue cones.

Achromatic versus chromatic channels

The achromatic channel has several characteristics that distinguish it qualitatively from the chromatic channels. Its responsivity is related linearly to the cone responsivities. It responds better to stimuli that vary quickly, both temporally and spatially. Thus, it responds to edges while the chromatic channels respond to areas. The achromatic channel extracts contrast from stimuli while the chromatic channels present absolute perceptions [4].

The perception of colour is not localized

Spatial and chromatic information are multiplexed on the same processing units when the opponent signals are created. Thus, the colour perceived at a given location depends on cone signals distributed throughout the stimulus. Because the spatial characteristics of achromatic and chromatic channels differ, the dependence of perceived colour on distributed input also varies. In general achromatic perception depends on more local variation than does chromatic perception.

Colour naming is surprisingly regular

Experiments with unique hues [2] show that colour naming is very consistent, both within and across observers, when the right naming protocol and visual conditions are established. The regularity of colour naming using ten or eleven colour names can be used as the basis for a natural categorization of colour [5].

Blue has extreme chromatic behaviour

Signals from blue cones seem to participate in a mechanism that has extreme chromatic behaviour. The spatial resolution of the blue part of the stimulus is limited by chromatic aberration; blue cones are sparse in the retina; and the yellow/blue channel has very limited spatial resolution. Taken together these indicate that distinctions mediated solely by blue cones, such as yellow-white and blue-black, are very poorly localized in space.

E. Summary

The last few sections illustrated several issues that crop up when colour is introduced into the workstation environment. The next few consider those issues in more detail, first those that occur within a single application, then those that involve the interaction of several applications, and finally those that are the special concern of the window system.

II. WITHIN AN APPLICATION

Within a single application, colour problems are similar to those that have existed in colour graphics for many years. Therefore the issues are relatively well-defined. It is surprising, however, to notice how few of these problems have clear cut solutions. Consequently, much of the discussion describes problems without giving more than hints at solutions.

Note that although each problem or capability is well-defined within a single application or window, it has a substantial system-wide component. Every example defines a capability that almost every application needs. Thus providing facilities on an application by application basis is extremely wasteful. Furthermore, many of the examples, including colour harmony and all the functional capabilities, produce significant interference when used in an environment where more than one application is visually present at the same time.

A. The aesthetic use of colour

This area of colour use has been little explored. It can be broken down into two categories. The first is the creation of screen designs that are aesthetically satisfying. The second is the preservation of aesthetically appealing features of images. The latter has been more popular in graphics research, largely because it seems more easily quantifiable. Nevertheless tools that help the users of colour displays to solve the problems inherent in finding an appealing set of colours are very much in demand.

1. Colour harmony

A user who decides to colour the elements of an application must choose many colours. Finding a harmonious set of colours is very difficult since it is equivalent to searching a space of high dimension, a task humans do not perform well. Existing colour selection techniques force him or her to make each choice independently of the others, which increases the difficulty.

Colour choice can be implemented in two ways. A colour choice facility can be integrated with the application. Alternatively it can be implemented as system software with colours communicated to the application when its execution begins. The first alternative makes choosing a set of colours easier since the feedback cycle is short. But it makes system-wide coordination of colour difficult to implement. The second alternative, while it makes colour coordination easier, and while it enforces consistency in colour choice technique, has a feedback cycle—choose colour set, start application, inspect colour combination, terminate execution—that is often so long as to be unendurable. (A third option is common in practice, of course. The application programmer codes colours into the application and the users live with his or her taste, like it or not.)

Design books with colour schemes that can be implemented as algorithms do exist (e.g. [6]). Colour selection techniques are well understood [7]. Putting the two together with better communication between system and application would make possible colour choice techniques that generate colour schemes on the basis of a small number of user choices. Once in place such tools would provide the basis for the experimentation needed to understand colour harmony better.

2. Windows spanning several display surfaces

Some workstations make it possible to have windows that span more than one display surface. Windows that do so offer the human visual system conditions for optimal discrimination of colour mismatches—large field, simultaneous presentation, small separation—and attention is focussed on the area in which the mismatch may occur. (In fact, the field of view is large enough that inhomogeneities in the visual field may be significant. Thus, if the CRTs have different phosphor sets a visual match may be impossible, regardless of calibration.) At the same time the CRT is judged on the weakest aspect of its performance, its colour balance. Techniques exist for calibrating CRTs and other display surfaces. [8] They can form the basis for tools to solve this problem. But the measurements require costly apparatus and skilled operators. Thus it is worth asking whether a restrictive policy of removing the display spanning capability might in fact be better.

3. Colour transfer between media

Applications that create colour images need to be able to print them satisfactorily. This problem is hard. Existing methods for colour printing [9, 10] require extensive measurement and some hand tuning on an image by image basis. Progress on automated colour printing is blocked on a shortage of scientific data showing what features of an image are important to its appearance.

B. The functional use of colour

Colour can be used to increase or reinforce the functional capability of an application. Current uses are based on a small set of properties of colour. They are all based on the non-locality of colour. (Non-locality means not that colour has no precise spatial location, because it has, but that the effects of colour are felt outside its precise spatial location.) Interestingly, this non-locality means that any one of these effects presents problems when used in a window system. These problems arise because the effects are predictable to the application only when the operations are confined to the area it controls.

1. Physical and conceptual properties of objects

Colour is a potent source of information about the physical properties of objects. In choosing food, for example, it is often the main differentiating factor between ripe and unripe. This power can be transferred to conceptual properties. When we associate the colour red, for example, with stop we are using it to indicate a conceptual property of the red object. Note that this effect is strongly context dependent: red means ripe for apples but yellow means ripe for bananas; green means edible for broccoli but green means inedible for meat. Similarly, conceptual properties can also be made context dependent. Red on a stop sign has a different meaning than red on a no stopping sign; the context is determined by the shape of the red area and its relationship to other visual objects. The ability to relate a colour to a conceptual property is the basis of colour coding.

2. Connectivity and grouping

Colour is an important factor in showing that two fragments are actually part of the same object. In a similar way the visual system is also able to assemble fragments of the same colour into a single visual group [11]. This ability is presumably based on the necessity of seeing different shapes as parts of a single occluded object. Colour is not the only visual property capable of grouping. Texture and motion, for example, also perform this function, but colour is particularly useful because grouping can be combined with colour coding. For example, seeing the highways or cities on a road map is usually assisted by grouping.

3. Visual search

Colour is very important as a cue in visual search. It is possible to search a large part of the visual field in parallel for objects of a given colour [12]. This effect can be used in two ways. First, it can provide a fast channel for time-critical communication, as occurs in the display of avionic information. Secondly, it can provide selective access to subsets of displayed information. This feature is important when the displayed information is very dense, as in maps and circuit layouts.

4. Commonalities

These examples of the functional use of colour have a great deal in common. It is easy to show that they work by designing displays that use them or by finding examples of effective displays that contain them. But there are few tools to help us. Research in avionics [13], for example, showed that the CIELUV uniform colour space provides a good basis for selecting colours for some tasks. If it is reasonable to assume that this result is generally applicable, we still face a significant calibration effort to choose colours based on it. Is there a simple technique that provides a good approximation with little calibration? Boynton's naming research[5] was based on the OSA colour system. Once again achieving exact colours depends on having an exact calibration available. But reality is even more complicated. Boynton did his research in an extremely simplified visual environment. How will the results change in the visually richer environment of the workstation CRT?

These and similar question, which exist for all the effects mentioned, indicate that a good prototyping environment [14] is most important if colour is to be well exploited. It requires software tools for aesthetic colour and an experimental methodology that makes it possible to derive engineering guidelines without the overhead of formal experimentation. Unfortunately the results generated in engineering experimentation are usually insufficient in generality to be used directly by other researchers. Thus, a further wrinkle on the needed methodology is sufficient commonality under differing experimental environments such that pooling of results will yield the generality that no single experiment can.

C. Functional and Aesthetic

When this topic is discussed the phrase is usually "functional versus aesthetic". The opposition is assumed to arise because purely aesthetic use of colour introduced by the user interferes with functional use of colour introduced by the programmer. For example, if the programmer chooses red as the selection colour, intending the user to find selected items quickly using parallel search, then any aesthetic use of red is assumed to interfere with the search. But does it? Clearly there are extreme colour usages in which it does. Choosing red for the background makes the selection completely invisible. There are also some usages in which it does not. Choosing a red title bar is unlikely to interfere. Between such obvious examples it is difficult to give guidelines. A better understanding of the psychological processes that underlie the various phenomena is badly needed.

In thinking about the problem, context should not be overlooked. A new visual context resets most colour effects. Aesthetic use of a colour in one context should not impede functional use of the colour in another context. But it is not possible to give a prescription for context and it is even unclear that visual context is a unitary phenomenon [15]. In general terms, context is created by visual distinctness. Thus, increasing visual context is gained at the cost of interface inconsistency. In particular, user control over colours within applications is likely to create context differences and inconsistency simultaneously. Intervention by the window system, described below, is likely to be a more attractive idea. A better understanding of the factors that create visual context and of the effect of context on the variety of colour functionality is important if the polarity between context and consistency is to be well used.

III. BETWEEN APPLICATIONS

A. Synchronizing colour hardware

Synchronizing the use of hardware by different applications is performed routinely by a wide variety of techniques. Some novel features appear, however, when we consider in detail synchronization of the colour map. The solution of choice is a static one, with the system finding a setting of the colour map that contains every colour desired by every application. Then applications can share the colour map without reference to the system. Such a strategy is usually successful only with full colour displays. For pseudocolour displays a dynamic synchronization strategy is required. When an application needs a new colour it requests a colour map entry and uses it. Commonly, however, the system runs out of entries, at which point this allocation strategy breaks down. Running out can be postponed if the system ensures that applications using the same colour, like black or white, use the same entry. The most efficient use of entries involves system monitoring of colours used by applications, with the system managing the colour map in whatever way maximizes the ability of applications to get the colours they need. The important point is that management of the colour map is best done when the system knows what colours each application needs, a small but important part of the content of its windows. We will see below that "running out of colours" is an even more severe problem perceptually since there tend to be many fewer usable colours than are commonly found in the colour map.

B. Synchronizing colour perceptions

There are a variety of ways in which colour used in one application can interfere with another application's use of colour. For functional colour the issues are at least understandable, if not soluble. For aesthetic colour they are much less concrete.

1. Functional colour

Some uses of functional colour rely on visual effects that explicitly utilize the ability of colour to act non-locally. Examples are visual search, creating connectivity and creating groupings. It is usually important that visual search should not find an object of the same colour but belonging to another application, and it is equally important that connectivity to or grouping with fragments created by different applications should not occur. The usual synchronization strategies exist. A static one allocates a particular property to each colour so that every object of the given colour is a target of the search, regardless of the application that created it. This sort of approach is also effective at enforcing consistency of colour use across applications. Unfortunately it comes up short because there are simply not enough colour categories. Dynamic strategies that allocate colours as needed are worse, since they give up consistency without decreasing the number of colour categories needed. Considering colour as a resource to be serialized is likely to be more effective. Thus, for example, only the active window might be allowed to use colour with all other windows displayed in grey scale. Of course, what is really being synchronized is the user's access to information mediated by colour. Therefore it may be possible to enhance performance by subdividing the user's colour associations. Manipulation of visual context provides the mechanism. If it is possible to organize the display so that each window acquires its own visual context then the search is confined to a single window. Currently rules for the creation of visual context are heuristic. Wide borders with strong visual texture and drop shadows, for example, are thought to be effective. Better research, leading to tools for the manipulation of visual context, is badly needed.

The opposition between context and consistency is important. When creating a new visual context we explicitly differentiate part of the display using visual texture, colour or any other visual cue. Because this differentiation weakens colour associations it is possible to build new ones. The new associations effectively reuse colour categories which are a scarce resource. Differentiation opposes consistency, which aims to give the system as a whole, including applications, a homogeneous interface to quicken learning and familiarity. Providing a visual environment in which applications can take advantage of both context and consistency is an important role of the graphic elements owned by the window system, as we will see in section IV.

2. Aesthetic

It is obvious that unsuitable neighbours can destroy the appeal of the most carefully constructed colour scheme. Two solutions to this problem are possible. The first is to coordinate colour choices throughout all applications. Doing so on an organized basis is likely to impose undesirable uniformity. But it is possible for the system to make small colour adjustments that bring the whole into a more desirable aesthetic balance. Such adjustment would require aesthetic principles embodied in algorithms and would be unobjectionable if done well. Similar adjustments of colours used functionally would, on the other hand, not be possible. The second solution to the problem of unsuitable neighbours is to use border elements like the frame of a picture to isolate the visual environment within a window. The use of border elements to create context also provides a reasonable solution for problems of interference between aesthetic colour in one window and functional colour in another.

IV. WITHIN THE SYSTEM

The window system should provide a set of tools that assist users and programmers in organizing displays and using colour. While helping to create a display that is aesthetically appealing and easy to use they should promote consistency in a visual environment in which each application is able to maintain a discrete visual context.

A. System-wide tools

Applications need tools that allow the user to create sets of colours. Using a tool supplied by the system—the colour picker in the Macintosh [16] system is an example of such a tool—has two advantages. First, the user sees the same interface each time he or she chooses colours, ensuring the advantages that consistency in the interface makes possible. This consistency has no cost in visual context since the method of choosing colours creates consistency while the colours chosen create the visual environment. Secondly, it may be possible for the system to use its role in colour selection for enforcing coordination of colours throughout the display.

When colours are chosen for functional reasons colour coordination becomes particularly important. There is an important interaction between functional and aesthetic colour that can only be mediated at the system level. Functional colours can be overloaded only if the applications using them use aesthetic colour to establish distinct visual contexts. Unfortunately, the scientific and design knowledge needed to build this capability into a window system is not available, but remains a significant challenge for future research.

The colour naming dictionary in X [1] is an example of this capability. It provides a system-wide interface to colour by a set of colour names. X also uses indirection to give a uniform interface within an application, referring to colours by their functional roles, foreground, background, and so on, which are understood by both system and application.

B. Visual elements for organizing the display

Significant portions of the display are owned by the window system itself, the desktop, title and scroll bars, window borders, and so on. These visual elements, which occur throughout the display, are an important aid when parsing the visual space into a set of discrete visual contexts. At the same time they unify it by their similarity [17]. Thus, when well designed, they are able to promote unity and diversity at the same time. It is interesting that X [1] allows the user to change these elements window by window, but users, in my observation, conservatively minimize inter-window variation.

The Macintosh [16] window system, by contrast, offers little choice to the user. The window elements are given a strong visual identity by their shape and texture This consistency is certainly important in making the interface design effective. This enforced consistency seems to be deeply rooted in the interface definition so that third party programmes for changing window colours, such as Colorizer [18], colour the elements of all windows consistently.

C. Rendering by the system

The logical culmination of the ideas presented above is a window system that receives very high level requests from applications to display one or another item of information, chooses rendering parameters, including colour, based on interactions with the user and internal design criteria, then renders the information to the display surface. Such a system is realizable, at least in principle, in systems such as X [1] that employ a monolithic server. The ability of most X applications to manage colour by indirection, such as foreground colour, is a step in this direction, as is the use of colour names. Unfortunately these values are not bound dynamically so that flexibility of system response to changing conditions is inhibited.

NeWS [19] is another approach to this problem. All applications communicate with renderers via Postscript. Because this higher level colour interface is interpreted, colour can be bound dynamically. It remains to be seen whether or not coordination methods can be developed that produce as much consistency as the global properties of colour demand without incurring the drawbacks of display-wide data structures.

V. CONCLUSIONS

A. Problems and Opportunities

The use of colour introduces a large variety of challenging new problems into research on workstation use. In the short run we need to find limits of colour difference that guarantee the efficient operation of capabilities like visual search, grouping and connectivity based on colour. It is similarly important to understand the operation of visual context so as to allow simultaneous use of colour for functional and aesthetic purposes. Solving problems like these ones requires us to acquire more information on how humans use colour for processing information, then to use that information when building tools and systems.

In the longer term problems centre on the interaction between the window system and applications that use the facilities it provides. The important non-local properties of colour force the application to give up control over details of colour rendering. Interestingly, issues similar to these colour problems occur in other aspects of the display, such as differing views of three dimensional objects. We may hope that the new concepts needed to handle colour well may offer a better substrate for solving those other display problems.

B. New Technologies on the Way

The advent of new technologies often creates a host of new ideas. We have discussed some problems and opportunities that occur as colour and window systems become more common in the workstation environment. Some of these new ideas concern the solution of problems of colour printing. The emergemence of colour printing as an important problem would be an interesting case study, since it has gone from an exotic curiosity to a problem of central interest in only a few years. Undoubtedly the availability of colour printers of reasonable quality and cost was important in this change. We now seem poised on the edge of another new technology of potentially great impact in colour graphics, the colour LCD display. Its different pixel structure compared to a CRT will raise a host of problems concerned with the microstructure of images, such as the best method for antialiasing coloured lines. The new opportunity of most relevance to the subjects addressed in this paper, however, is the dual identity of the LCD as either self-luminous or reflective display surface. Most current representations of colour in computer graphics are based on the premise of a self-luminous display. Thus, accommodation of LCD displays might require a drastic rethinking of the lowest level colour representations. The consequences of such a change would, of course, ripple through all the topics addressed here.

VI. ACKNOWLEDGMENTS

The research discussed in this paper has been partially supported by grants from the National Science and Engineering Research Council of Canada and from Apple Canada. Much of this material formed part of the Advanced Computer Graphics course at the University of Waterloo and I would like to thank the students for many interesting discussions on these topics. Finally, I would like to thank Tanis Day for stylistic comments and Marc Gauthier for the translation of the abstract.

VII. REFERENCES

1. Oliver Jones, *Introduction to the X Window System*, Prentice Hall: Englewood Cliffs, 1989.
2. Robert M. Boynton, *Human Color Vision,* Holt Reinhart & Winston: New York, 1979.

3. There is some disagreement as to whether or not the blue cone contributes to the achromatic channel.
4. Michael von Grünau and William Cowan, "The role of simultaneous contrast in achromatic and chromatic channels", *Investigative Ophthalmology and Visual Science,* 29, 301, 1988.
5. Robert M. Boynton and Conrad X. Olson, "Locating Basic Colors in the OSA Space", *Color Research and Application,* 12, 94–105, 1987.
6. Wucius Wong, *Principles of Color Design,* Van Nostrand Reinhold: New York, 1987.
7. Michael W. Schwarz, William B. Cowan and John C. Beatty, "An Experimental Comparison of RGB, YIQ, LAB, HSV and Opponent Colour Models", *ACM Transactions on Graphics,* 6, 123–158, 1987.
8. William Cowan, *Colorimetric Properties of Video Displays,* Course notes, Optical Society of America Annual Meeting, Rochester, 1987.
9. Maureen C. Stone, William B. Cowan and John C. Beatty, "Color Gamut Mapping and the Printing of Digital Color Images", *ACM Transactions on Graphics,* 7, 249–292, 1988.
10. Michael G. Lamming and Warren L. Rhodes, "A Simple Method for Improved Color Printing of Monitor Images", to appear in *ACM Transactions on Graphics.*
11. Colin Ware and John C. Beatty, "Using Color Dimensions to Display Data Dimensions", Human Factors, 30, 127–142, 1988.
12. Ellen Carter and Robert Carter, "Colour and Conspicuousness", *Journal of the Optical Society of America,* 71, 723–729, 1981; Richard Christ, "Review and Analysis of Color Coding Research for Visual Displays", *Human Factors,* 17, 542–570, 1975.
13. Louis D. Silverstein and Robin M. Merrifield, *The Development and Evaluation of Color Systems for Airborne Applications,* U.S. Department of Transportation, Federal Aviation Administration, 1985.
14. Lyn R. Bartram, Kellogg S. Booth, William B. Cowan and Peter P. Tanner, "A system for conducting experiments concerning human factors in interactive graphics", *Proceedings of Graphics Interface' 88,* 34–42, 1988.
15. A. Triesman and H. Schmidt, "Illusory Conjunctions in the Perception of Objects", *Cognitive Psychology,* 14, 107–141, 1982.
16. Macintosh is a registered trade mark of Apple Computer, Inc.
17. Robert Venturi, Denise Scott Brown and Steven Isenour, *Learning from Las Vegas,* revised edition, MIT Press: Cambridge, 1977.
18. Colorizer is a Macintosh application program marketed by Palomar Software, Inc.
19. *NeWS Preliminary Technical Overview,* Sun Microsystems, Inc., Mountain View, California, 1986.