A VIDEO TERMINAL FOR INTERACTIVE GRAPHICS

N. Burtkyk*, P. Carey†, K. Steele*, and M. Wein*
*National Research Council of Canada, Ottawa
†Lektromedia Ltd., Montréal

Abstract

A video frame buffer display terminal with full graphics capability is described. It provides a high level of user support that is suitable for a variety of interactive graphic applications. It uses a flexible picture data structure which can be created, displayed and manipulated within the terminal under user control through menu selection. Variations in the characteristics of this graphics operating environment are controlled by the host-resident application program which defines these menu functions.

TERMINAL VIDÉO POUR SYSTÈMES GRAPHIQUES INTERACTIFS

Résumé

Un terminal vidéo à mémoire permettant une gamme complète de traitements graphiques est décrit. Le soutien fourni à l'utilisateur par le terminal est tel qu'il rend possible toute une variété d'applications graphiques interactives. Le terminal utilise une structure de données graphiques souple qui peut être créée, affichée et manipulée par le terminal sous la gouverne de l'utilisateur, grâce à une sélection de fonctions. Les modifications pouvant être apportées aux caractéristiques de ce système de traitement graphique sont régies par le programme d'application résidant dans l'ordinateur principal, lequel définit les fonctions à sélectionner.

A VIDEO TERMINAL FOR INTERACTIVE GRAPHICS

Introduction

The variety of graphics system concepts that have been employed for interactive applications is nearly as broad as the applications themselves. The early potential of dynamic graphics was best illustrated in stand-alone refresh graphic systems where the full processing capability of the central processor was available and dedicated to a given application. Although the costs of dedicated systems have been decreasing continually, they have remained out of reach for all but very specialized applications.

The most rapid growth in practical graphics has been through the medium of the storage tube terminal where, for a modest investment, graphical communication is accessible to the user of a shared central computer resource. Its main advantageous characteristic is, of course, also its most serious limitation - a storage tube which needs no refresh is essentially static. The sophistication of such terminals has grown in certain respects, but without any local processing power all user interaction necessitates data transfers up and down the line. In spite of this , many interactive applications have been developed, with their performance depending mainly on the response delay from the shared host computer.

At the other end of the spectrum of terminal sophistication is a fully dedicated refresh graphics system connected to a large host computer as a satellite system. The application program is automatically distributed between the two processors which are connected by a wide band line in an attempt to make the resources of the host computer available whenever necessary (1,2,3). Proponents of such systems have promoted this concept as a combination of power and flexibility, but since it does little to offer economy, the investigation of such techniques remains mainly an academic exercise.

The development of the TEKTRONIX 4081 graphics system represents one significant approach to affordable graphics based on the "write-thru" feature of a storage tube (4). It circumvents the limitations of the simple storage tube terminal while retaining its advantages. However, the division of the application program between the 4081 and the host computer remains difficult and the system is most conveniently used in a stand-alone configuration.

Another approach originates with the video character terminal. Since the advent of the microprocessor, character terminal technology has undergone significant evolution which presently includes a capability for limited graphics (5,6). An enhancement of the video terminal to include full graphics capability has recently been reported (7). The major objective was to overcome the limitations of a storage tube terminal through the use of a frame buffer memory with selective erase. To that extent, the project was successful, but it represents only an initial step which gains significant advantage mainly in text processing applications. An extension of this approach through the development of an integrated system of graphics hardware and software offers a promising technique for economical yet powerful graphics.

This paper describes a video terminal graphics system which is being developed under a joint industry/laboratory program between LEKTROMEDIA LTD. and NRC. Our goal was to design an advanced graphics tool which is controllable by an application program in a natural and simple way. As a graphics system, it provides a high level of user support that is suitable for a variety of interactive graphics applications.

System Concept

The primary objective in this development was to achieve a low cost video display terminal that is suitable for general purpose graphics applications. The implication of general purpose support was to provide a highly user-oriented graphics environment to which an application program can interface easily. The hardware system is based on a microprocessor driven multi-plane frame buffer memory with flexible write capability. Various supporting hardware features are included to achieve an effective overall system implementation. The integrated system concept is augmented by a versatile software package for graphic interaction.

The conversational process of interactive graphical communication between man and the computer is represented in Fig. 1. It should be stressed that this addresses man the user and not man the programmer. The input interface provides the user with a means for constructing and manipulating a graphical data structure, while the output interface presents pictorially the current status of this structure. Constraints that are applied during the constructive and interpretive phases of this process reflect the characteristics of the specific application program.

In a stand-alone graphics configuration, the important functional boundary is at the user interface, since all components of this system reside within the same computer. For a terminal configuration, a second physical boundary at the line interface exists. It is desirable to make this boundary appear transparent to the user. Although there is sufficient local processing power to support user interaction, ease of programming makes it desirable to locate the application program entirely within the host computer. Thus the support package within the terminal should include only

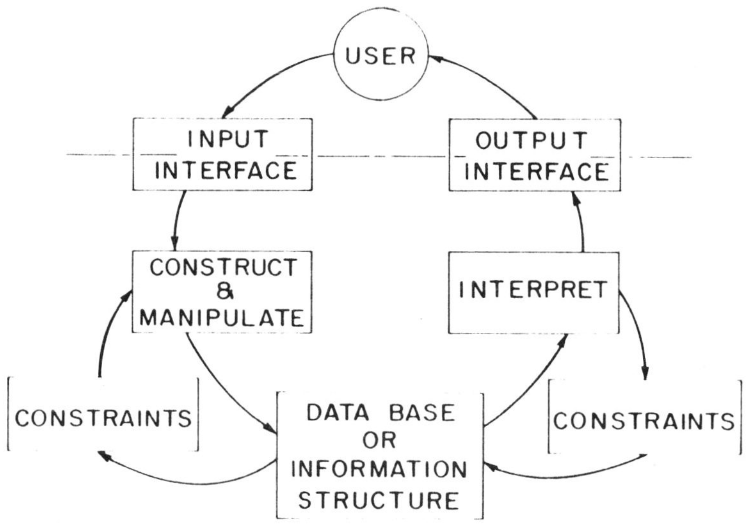

Fig. 1. - Diagram of the essential elements of interactive graphics.

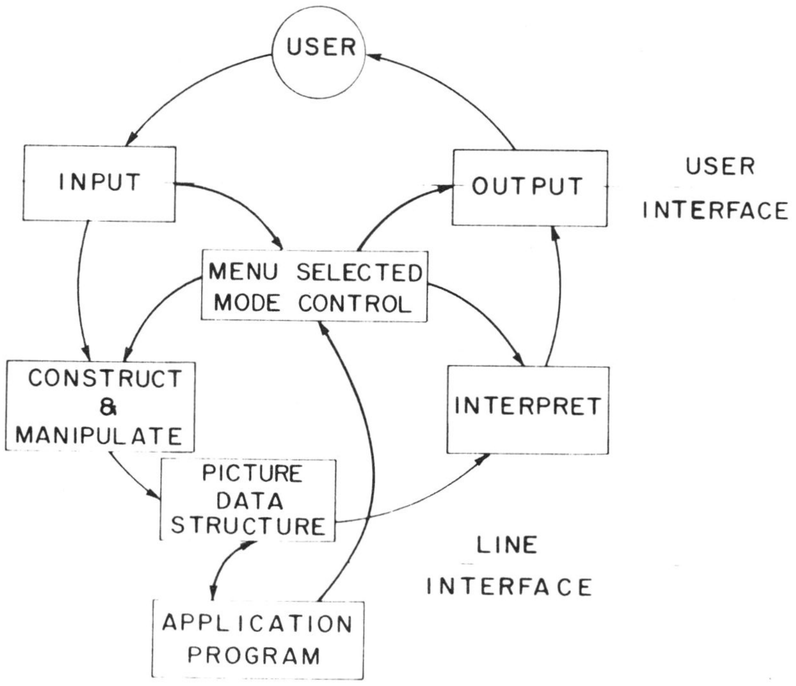

Fig. 2. - Graphical interaction in an intelligent terminal configuration.

those interactive capabilities which are common to graphics applications.

The model of the user interactive process is now redrawn as in Fig. 2, where the host-resident application program has been identified and separated. All the essential tools for graphical interaction have been retained at the user interface but in such a form that it translates into meaningful communication with a remote application program. At the user level, he can create, manipulate and control the displayed images locally at the terminal through menu selection. At the application program level, the particular characteristics of this graphics operating environment are controlled through the definition of the menus which are presented for user selection. This application program/user control interchange permits a natural dialogue between the user and the system which makes the line interface largely transparent.

Hardware Configuration

The basic hardware configuration of the terminal is given in Fig. 3. The graphics subsystem consists of a microcomputer, a digital frame buffer, a frame buffer controller, a video generator and a TV monitor. Refresh of the video display is generated continuously from the frame buffer. This frame buffer is a random access memory configured as 480 lines of 640 points per line to provide the visible portion of a standard 525 line interlaced video signal. The communications subsystem consists of a second microcomputer with I/O interfaces for communicating with the host computer and with local peripheral devices. One of these devices is a touch panel which overlays the display monitor and permits user interaction by pointing on the screen.

The frame buffer memory is constructed in planes of 16 bit words, each plane containing one bit for each raster point or pixel in the displayed image. During writing, these buffer planes may be written in parallel for grey scale presentation or selectively for overlayed foreground/background separation and keying. The foreground/background capability permits a foreground image to be manipulated locally within the terminal while a complex background image, transmitted only once from the host computer, remains undisturbed. During video generation, data is always read in parallel from all buffer planes and combined in various ways to produce greyscale, overlay, keying or any other logical operation. Using the maximum of three planes permits a limited capability for color generation.

The controller contains all the address and timing logic for writing graphics information into the frame buffer. Line generation is provided by a binary rate multiplier vector generator which sets all the intermediate points between specified endpoints of a line. This write process is completely interleaved with the video readout, one write or read/modify/write cycle being permitted between each read access. A diagonal line traversing the screen is written in about 1 ms, since only one raster point can be set during

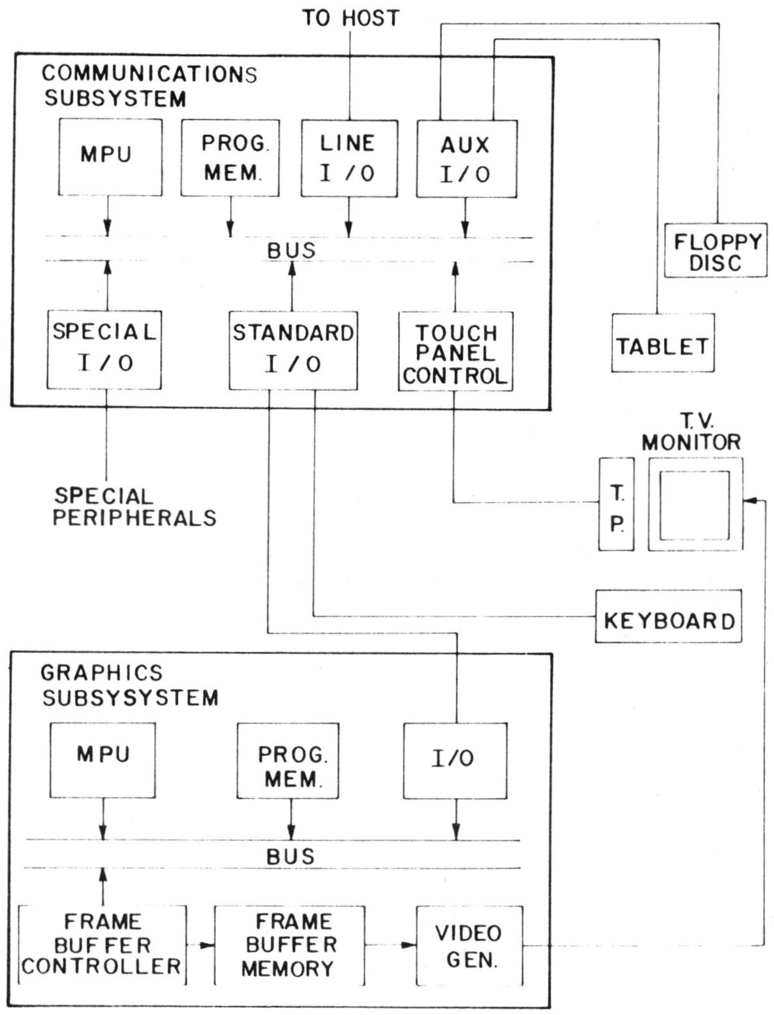

Fig. 3. - Hardware configuration of the video graphics terminal.

each memory access. For horizontal lines, successive bits in the same word are set in one memory access, so the time to traverse the screen is only 100 usec. This permits rapid generation of solid rectangular areas of display. The line generating hardware operates over a 12 bit range with off-screen blanking controlled by limit registers. These registers are loaded under program control and can provide any viewport within the viewable area of the display screen.

The generation of alphanumeric characters is also provided through this line drawing interface. Although based on a 7 x 9 dot matrix description, the character ROM's actually contain incremental vector encoding which permits characters to be scaled and rotated before being presented to the line generator for display For normal upright characters, the writing speed exceeds the transfer rate on a 9600 baud line. Typically, 32 lines of text may be presented on the screen with excellent legibility.

All the input and control devices communicate with the processor in the communications subsystem. In addition to the standard devices (keyboard and touch panel), peripheral interfaces are available to support a data tablet and a floppy disc drive. For special applications such as computer aided instruction, audio and slide projector control interfaces are also available.

Software System

A functional block diagram of the terminal graphics software is shown in Fig. 4. The essential components for local interaction are the menu display and execution routine, the graphical data structure and the group of interactive routines that operate on the data structure. The graphics interpreter is active only during graphic data transfers between the terminal and the host resident application program.

The form of the data structure is given in Fig. 5. Picture components or entities are queued in a picture buffer. Each entity consists of a header or control block and a data block. The control block specifies the current graphic transformations (positional offset, scale, rotation, viewport) which will be applied to the data block during display generation. The data block may contain coordinate data for vector endpoints or ASCII strings for character display. Each of these entities is referenced in an entity table which contains its I.D., its active status and an address pointer to the picture buffer. Access to the data structure is always through a management routine which maintains and updates all address pointers while scanning through the entity table.

Entities exist in four distinct forms. PICTURE and MESSAGE entities have full control blocks and can be independently manipulated. SUBPICTURES and LABELS have no control blocks but their displayed state is determined by the control block of the PICTURE which references them. This structure permits a completely free mixture of line graphic and character display with moderate versatility and

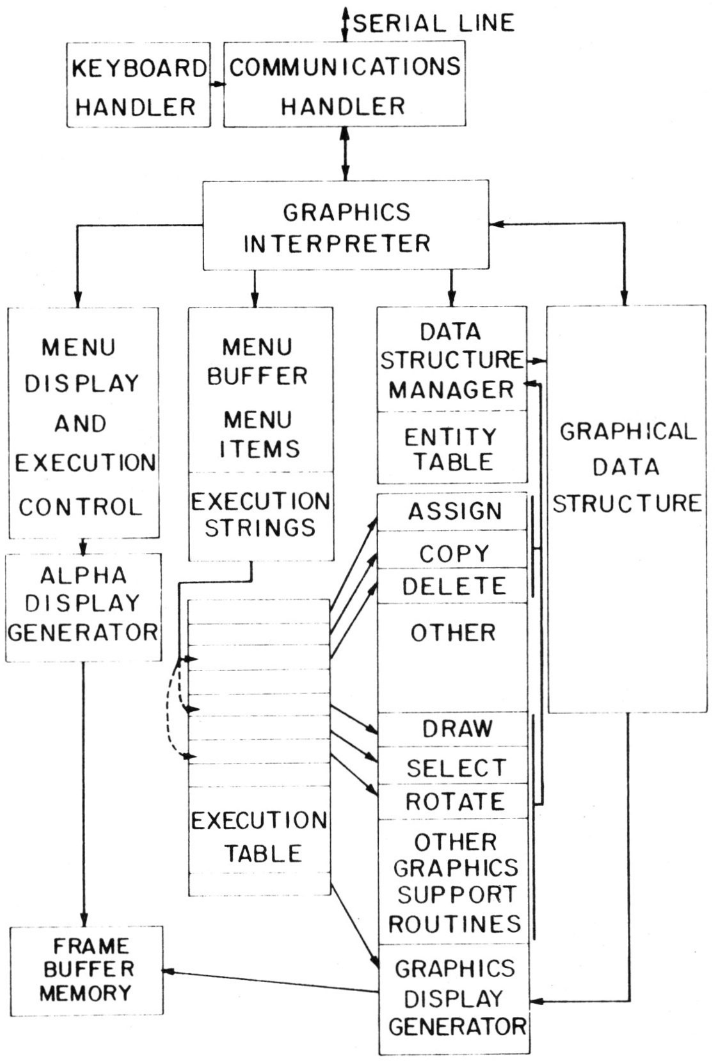

Fig. 4. - Functional diagram of the terminal graphics software.

flexibility for user interaction.

Graphic interaction is achieved through the controlled execution of functional routines that operate on this data structure. This control is provided through a menu facility which presents a list of items on the display for user selection. Associated with each item of a menu is an execution command string which specifies a sequence of graphic operations to be executed. Each string element is a pointer to a functional routine referenced via an execution table. The last pointer in this string specifies where control will return upon completion. For the stand alone mode, internally defined menus are provided to control the interactive process. When connected to a host computer, the application program provides its own menu definitions for control. The nature of the functional routines that may be invoked through menu control characterizes the capabilities of the system.

The primary routine is a graphics display generator which processes the data in the picture buffer and updates the display. This display generator does not create a stored display list as in conventional refresh graphics. Instead, as each display instruction is created, it is passed to the frame buffer where it is converted into a raster bit map. The display generator is invoked only when the display status of the picture data structure has been modified.

A second major routine permits a user to draw or construct a picture entity by adding new point coordinates to a data block. Associated with this is a data management function to assign and initialize new space in the picture buffer. Other companion routines include SEARCH, COPY, DELETE, EXTEND, COMPRESS, etc. for additional management functions. SELECT, for selecting an entity on the display by pointing at it, and MOVE, ROTATE, SCALE, form a set of manipulation functions for manipulating a selected entity using a specified control device. Continuous manipulation of a picture involves continuous looping between the manipulation and display generator routine, one modifying the parameter values in the control block and the other updating the display to reflect the change. The foreground/background buffer plane separation is particularly useful in this mode for maximizing response speed by reserving the foreground display for the picture being manipulated.

Communication with the application program involves additional terminal resident routines for instruction and data transfer. These capabilities are represented functionally by the Graphics Interpreter block of Fig. 4. Picture data and menu files which are received from the application program must be decoded, formatted and routed to the proper buffer. Picture data and user requests must be accessed and encoded for transmission to the application program. These functions are all initiated at the terminal through menu selection. Similarly, any processing in the host occurs only as a result of menu selection which returns control to the host. Information transmitted from the terminal to the host is primarily:

1) Menu selection returns
2) Picture data
3) Other data (position coordinates, character strings)
4) Error condition.

Information transmitted from the host to the terminal is primarily:

1) Picture data
2) Menus
3) Commands and prompts
4) Error condition.

This information flow at the host is indicated in Fig. 6. The host does not normally issue commands other than for clearing or deleting entities from the terminal picture and menu buffers or to set the modal operating conditions for the terminal. In general such commands are required or their use implied as a consequence of a particular menu selection.

Information exchange between the host and terminal is extended beyond solely menu selection by providing a prompt capability from the host. This allows the host to transmit text messages to be displayed at the terminal for the information of the user. These messages may be requests for information to be entered at the terminal keyboard or for position data from one of the pointing devices or all of these. This prompt facility is always used within the host in conjunction with some processing mode specified by a previous menu selection. It cannot be used instead of a menu selection since it always originates at the host. Prompts allow flexibility in that various conditions may be in effect at the time a particular menu selection is made and these conditions are in general only known in detail to the host application program.

Whenever control is returned to the application program, identification of the menu item which was selected is transmitted together with any associated data. For example, if a SAVE function is selected, the execution sequence would consist of a keyboard entry to identify the name of the picture entity being saved, followed by a transfer of the associated picture data from the picture buffer together with identification of the selected menu item. Upon receiving this data, the application program recognizes that the SAVE function has been selected and completes the process by accepting the data. After completion, the application program may command the terminal into a new mode, or it may simply transmit an acknowledgement which signals the terminal to continue in its previous mode waiting for further user interaction.

The effective graphics capability at the line interface is translated to the application program interface by a corresponding host resident support package. This is a set of Fortran compatible routines which make the terminal appear as an extension of the host computer. It accepts and returns picture data. It transmits message prompts for user guidance and accepts user responses or requests. It transmits menu definitions for user control and commands

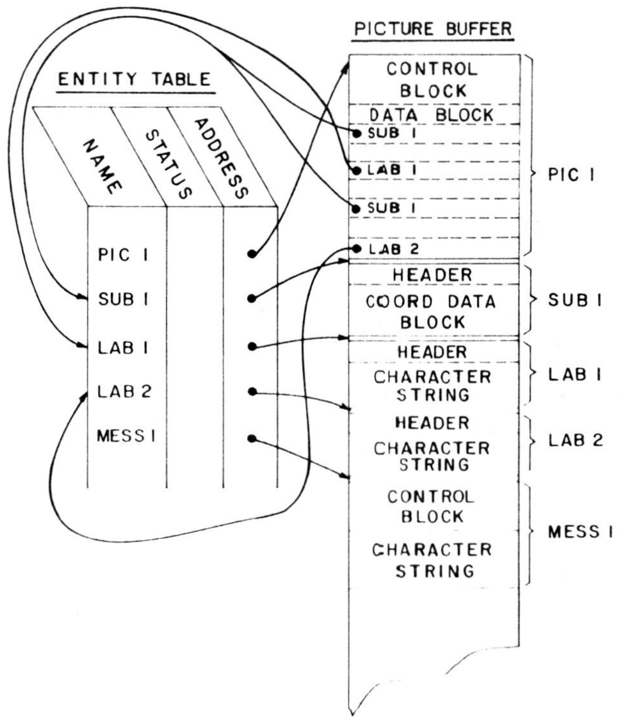

Fig. 5 - Picture data structure.

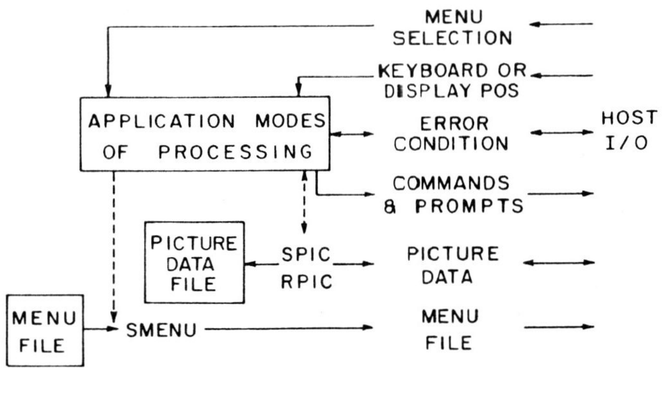

Fig. 6 - Information flow between the host CPU and terminal.

for menu activation. A summary of the essential subroutines that make up this graphics support package are listed in Table I.

Table I - Host Support Routines

High level calls

SPIC,RPIC	- Output/input picture data
SMENU	- Output a menu
SDIR	- Output picture data for direct display
PROMPT	- Output text and optionally request keyboard and/or display position information

Low level calls

PREDY	- Host ready to accept input
PBIN,GBIN	- Output/input binary data
PVEC,GVEC	- Output/input vector data
PKAR,GKAR	- Output/input character data
PCOM	- Output command
PDVEC	- Output vector for direct display
PDKAR	- Output character string for direct display

The graphics support package contains two levels of calls. At the highest level, full support is provided for a data structure that is identical to the format used in the terminal. When a picture is sent from the terminal, a simple call will accept it and regenerate the standard format. If a special format is to be used, then multiple calls must be made at the lower level to pick up individual coordinate pairs and formatting becomes the responsibility of the application program. An equivalent process is involved in transmitting a picture to the terminal for manipulation and display. If, however, a picture which exceeds the picture buffer storage capability of the terminal is to be transmitted to be used as a background image without any manipulation, it is processed into a display instruction set for direct display as with a storage tube terminal. Again only a single high level call is required if the standard format is used. Otherwise a low level call must be made to output each individual vector as it is generated in the application routine.

Conclusion

The design of this terminal has been based on techniques and experiences gained in previous work in developing stand-alone graphics applications. Much of the design emphasis was placed on making the terminal appear to behave like a stand-alone graphics system. This design goal has in the main been achieved, but with one noticeable exception. In a stand-alone configuration, the user is unaware of the distinction between those operations which involve frequent data transfers between the application program and the graphics support environment, and those which do not. Under our terminal concept, these two components of software are separated by a serial communications in which special processing is required for each user interaction, the performance tends to degrade to that of a conventional non-intelligent terminal. Nevertheless, this terminal concept offers substantial benefits to a wide range of graphic applications because of its local capability for manipulating and interacting with the display.

For an extended terminal configuration including a floppy disk, the provision of a simple picture data storage capability would be attractive. Apart from convenience, this would enhance the overall performance since recovery of picture buffer space would be permitted without having to transfer picture data to the host for storage. In addition, it would allow extension of the repertoire of executable graphics functions beyond the available memory capacity of the terminal. This extended repertoire of executable graphics functions beyond the available memory capacity of the terminal. This extended repertoire could even include alternates to the standard basic routines if required for any specialized applications.

Acknowledgement

The authors wish to acknowledge the valuable assistance provided by Mr. Gasoi, Mr. Levesque, and Mr. Friedlander of LEKTROMEDIA LTD particularly in the hardware design aspects of this development.

References

1. A. Van Dam and G. M. Stabler, "Intelligent Satellites for Interactive Graphics", AFIPS Conf. Proc., 1973 Nat. Computer Conf. Vol. 42, pp 229-238.
2. J. D. Foley, "Software for Satellite Graphics Systems" Proc. of the ACM 1973 Conf., pp 76-80.
3. J. D. Foley, "A Tutorial on Satellite Graphics Systems", Computer, Vol. 9, No. 8, Aug. 1976, pp. 14-21.
4. N. Thanhouser, "Intermixing Refresh and Direct View Storage Graphics", Computer Graphics, Proc. of SIGGRAPH' 76, Vol. 10, No. 2, pp 13-18.
5. Hewlett Packard, H.P.-2640 terminal.
6. Intelligent Systems Corp. Intecolor 8001 terminal.
7. F. Baskett and L. Shustek, "The Design of a Low Cost Video Graphics Terminal", Computer Graphics, Proc. of SIGGRAPH '76, Vol. 10, No. 2, pp 235-240.