第13卷 第6期
1987 年 11 月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 13, No. 6
Nov., 1987

微计算机控制的多画面交互式彩色图形系统

郑 君 兰
(中国纺织大学)

摘 要

本文阐述一个低成本、应用灵活的交互式彩色图形系统。它包含一个用于画面前后台显示处理的多画面存贮管理部件，具有彩色图形、字符、伪彩色图象和动画四种显示模式。系统的交互功能(例如图象/图形的输入，修改与画面编辑等)由系统软件和一个易于扩充的图形命令语言定义，因此可望该系统能在多方面获得应用。

纵观计算机图形技术的发展史，系统的成本始终是影响其应用范围的主要因素。长期以来，研制新的图形硬件与软件，使系统价格降至大多数应用可以接受的水平，一直成了人们寻求的目标[1,2]。至七十年代后期，计算机图形的应用仍主要限于学术研究部门[3]。最近几年，由于图形系统成本逐年下降，特别是微计算机图形系统的出现，其应用范围开始扩展到数据处理工业中的各个领域[4]。

为适应这种发展趋势和国内亦在增长的应用需求，本文阐述一个微计算机交互式图形系统的设计与实现。该系统是针对实际应用环境而研制的，目前已在纺织品的 CAD 中获得应用。

一、系统硬件结构

系统由一台经过改进的国产八位微计算机(包括软磁盘、打印机、键盘等通用外部设备)和自行研制的图象输入与显示装置、具有模块结构的画面存贮器及其管理部件组成(图 1)。其主要特征是采用了一种可以在微计算机与显示控制器之间灵活组织与共享多个画面存贮模块的结构。微计算机可以通过存贮管理部件对任一画面模块直接寻址；与此同时，显示控制器亦可选择任一画面模块作为自身的帧缓冲存贮器显示该画面。这种共享存贮资源的结构使系统 CPU 同时兼有 DPU 的职能，无须附加专用硬件接口在计算机与显示帧缓冲存贮器之间传送数据，降低了系统成本，提高了画面显示的响应速度。

1.1. 多画面图形存贮与管理

系统图形存贮器具有可扩充的模块结构。每一存贮模块由四个 256 × 256 × 1 bit

本文于 1985 年 7 月 19 日收到。

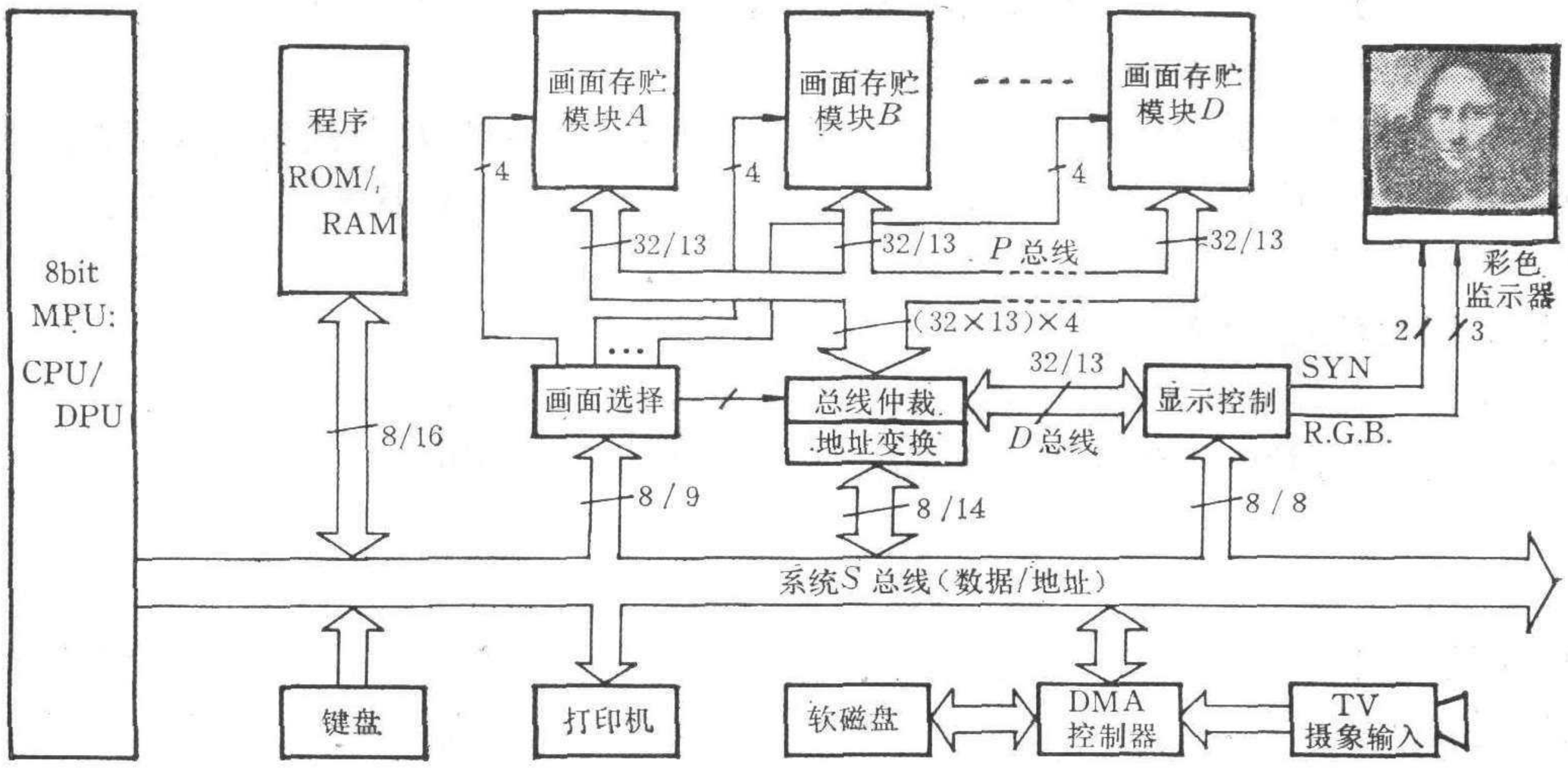

图 1　系统硬件结构

的位平面组成。模块内部或模块间可以灵活组织。例如用一个模块的四个位平面存贮一幅 256 × 256 象素、十六种彩色的图形，或用四个模块存贮一幅 512 × 512 象素的图形，也可用其中三个位平面存贮八种彩色的图形，另一位平面作为该图形的标志（mark）平面。

图形存贮器受多画面存贮管理部件的控制，该部件在计算机系统与显示控制器之间进行画面存贮分配，为此每个画面存贮模块设有各自的局部 P 总线，分别和系统 S 总线或显示控制器的 D 总线连接。 CPU 将一个画面控制字装入存贮管理部件中的画面选择寄存器，即可同时确定两个(或同一个)执行前后台操作的存贮模块。

当某一画面存贮模块位于后台时，其地址 A_p 由 P-S 总线映射为微计算机寻址空间中的 A_s

$$A_s = A_p + M_0 \quad (A_p = 0, 1, \cdots, M-1). \tag{1}$$

式中 M 为单个画面存贮模块的容量，M_0 为地址映射初值，它由包含最高有效地址位的画面选择寄存器确定。

位于前台的显示图形数据由 P-D 总线送显示控制器。 由于图形存贮器读写周期的限制，数据传送需要采用并行方式来满足视频显示的要求。 对于 D 总线上的某个地址 A_d，可以同时读出 P 总线上 n 个地址间隔为 δ 的数据

$$A_p[i] = A_d + (i-1) * \delta \ (i = 1, 2, \cdots, n). \tag{2}$$

式中 δ 满足关系式 $\delta = M/n$. 因此当 A_d 由 0 计数至 $\delta - 1$ 时，即可完成整幅画面数据的传送。采用这种方法，存贮器中的图形数据需按特定规则排列才能获得正常的显示画面。显然，当显示画面退居后台时，如果仍然按照式(1)实现 P-S 总线的地址映射，图形数据的不连续地址将给数据处理带来困难，为此本系统对 CPU 访问画面存贮模块的相对地址 $A'_s(=A_s - M_0)$ 进行下述变换：

$$A_p = \mathrm{INT}(A'_s/n) + (A'_s - \mathrm{INT}(A'_s/n) * n) * \delta. \tag{3}$$

这一变换(式中 INT 表示取整运算)将使画面存贮模块中图形数据的特殊排列对程序设

计者是透明的。实际上，若 n 取 2 的整数幂值，式(3)的变换很容易用硬件实现，它相当于将 A'_i 的二进制值循环右移 $\ln n$ 位。

当画面的显示与处理操作在同一模块中进行时，将产生 S, D 总线争夺局部 P 总线的情况。为此设置了一个总线仲裁部件，用于管理与控制上述竞争状态下总线上的信息流，确定 CPU 与显示控制器访问画面存贮器的优先权。

1.2. 图象/图形输入

系统用一台标准制式的电视摄象机作为图象输入设备，通过一个高速 DMA 通道传送二值影象数据，数字图象 (G) 用二值影象叠加的方法获得[5]

$$G = \sum_{k=0}^{m-1} (T_k - T_{k-1}) B_k. \tag{4}$$

式中 B_k 是对应灰度阈值 T_k 的二值影象。选定 $m-1$ 个阈值，即可在输入过程中将原始图象分割成 m 级灰度，因此该装置又可同时用作图形数据输入设备。在输入位置精度要求较高的场合，可用键盘操纵光标修正画面数据，或直接用图形命令控制光标作图。

1.3. 图形显示

系统的显示输出部分具有视频传输管路 (Video Pipeline) 结构，它包括一个环形数据缓冲器、译码器和一个彩色查找表 (Colour Lookup Table) (图 2)。

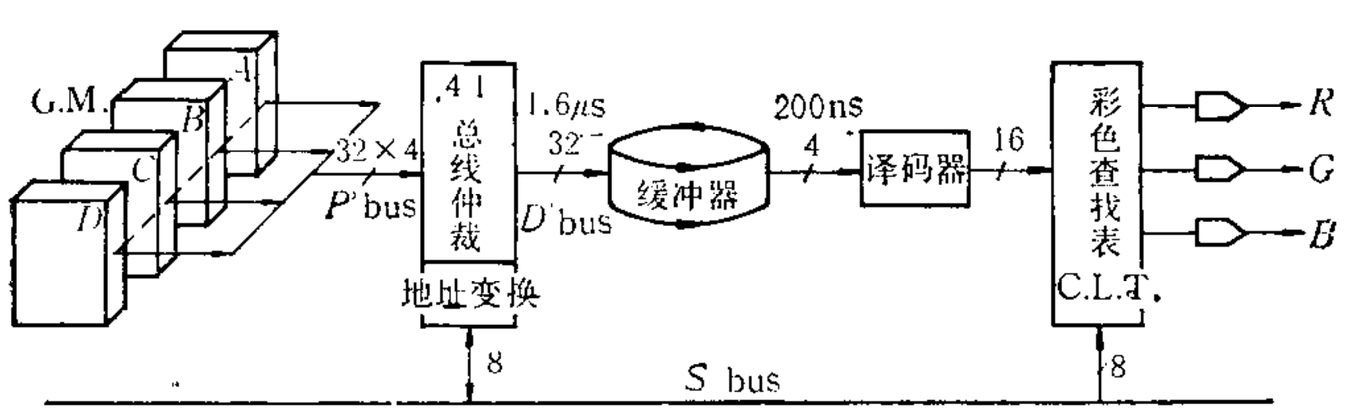

图 2 视频传输管路 (Video Pipeline) 结构

环形缓冲器用于匹配管路中的信息传输速率。它并行接收来自画面存贮模块的数据，以视频传输速率串行输出。管路输出端是一个 16 × 12 bit 的彩色查找表 $C = (C_0, C_1, \cdots, C_{15})$，它确定了十六种当前屏幕显示的彩色 (或灰度)。缓冲器输送的图形数据 $i[i \in (0, 1, \cdots, 15)]$ 由查找表变换为彩色码 C_i 送监视器屏幕显示。

二、显 示 模 式

灵活地组织与分配画面存贮模块，配合适当的标色技术，系统能够以彩色图形、字符、伪彩色图象和动画等四种显示模式工作，满足不同类型的应用要求。

2.1. 彩色图形/字符模式

每一图形/字符画面由存贮模块的三个位平面组成，能够同时显示八种不同彩色。另一位平面(高位)用作该画面的标志面，显示图形光标，画面裁剪窗口，字符游标以及图形变换的辅助线等。以这种模式工作时，与标志面对应的彩色查找表元素设置为

$$C_i = C_m \quad (i = 8, 9, \cdots, 15). \tag{5}$$

式中 C_m 为标志面图形彩色，它可以在八种画面图形彩色（C_0—C_7）之外任选。

2.2. 彩色图形、字符交替模式

这种显示模式与图形/字符模式的区别在于画面的标志面被独立地用作字符显示，它具有通常 CRT 字符显示器的文本编辑(例如插入、删除、滚行等)功能。

当彩色查找表按式(5) 设置时，C_m 指定了字符显示器的彩色，可以获得单色字符与彩色图形背景重叠显示效果，适用于需给某一图形画面加上“字幕”说明的场合。若将彩色查找表中对应标志面的各元素修改为

$$C_i = C_{i-8} \quad (i = 8, 9, \cdots, 15), \tag{6}$$

则重叠显示的字符变为透明色，字幕消隐而仅显示彩色图形画面；如需将彩色图形消隐，重显字符，可以重新定义彩色查找表

$$C_i = \begin{cases} C_n & (i = 0, 1, \cdots, 7), \\ C_m & (i = 8, 9, \cdots, 15). \end{cases} \tag{7}$$

此时彩色图形合并为 C_n 背景色，屏幕被作为单色 CRT 字符显示器使用。因此利用式(5),(6),(7)，可以相应得到彩色图形、字符的重叠或交替显示。

2.3. 伪彩色图象模式

系统以这种模式工作时，相当于一个小型数字图象处理系统。每幅图象由画面存贮模块中的四个位平面组成，因而能同时以十六种不同彩色 $C_i(i = 0, 1, \cdots, 15)$ 显示由电视摄象输入装置获取的十六灰度级数字图象。改变 C_i 即改变了相应灰度级的彩色。若使 $C_i = C_j$，则图象的第 i, j 灰度级被合并，用这种方法能够实现图象的密度分层显示。

2.4. 动画模式

系统在选择某一画面进行(前台)显示的同时可对其它画面进行(后台)处理。如按一定时间进行前后台画面切换，则可产生动画显示。当动画序列涉及到整块面积的修改时，采用上述画面切换方式产生动画是有效的，主要的限制因素是构造动画序列的算法及处理机的速度。对于复杂的动画，可将生成的序列存放在不同的画面存贮模块中，系统能按照程序设定的时间间隔自动切换显示画面，获得不同速度的动画演示效果。

三、系统软件——GCL 图形命令语言

GCL (Graphic Commad Language) 是为本系统设计的一种通用的图形命令语言。它为系统与各种应用环境之间提供了一个统一的界面，用以实现交互图形系统通常应具备的典型功能。

3.1. 结构

GCL 具有树形分层结构，它可以包含任意多个子树，组成分枝形图形命令系统（图3)。

GCL 中的每一图形命令都由路径名 (Path Name) 标识。命令有两种格式：

〈**路径名**〉|〈**路径名**〉〈参数〉〈*CR*〉。

路径名定义为命令树中节点名的有序集合。指定方法是从根开始沿树的节点跟踪命

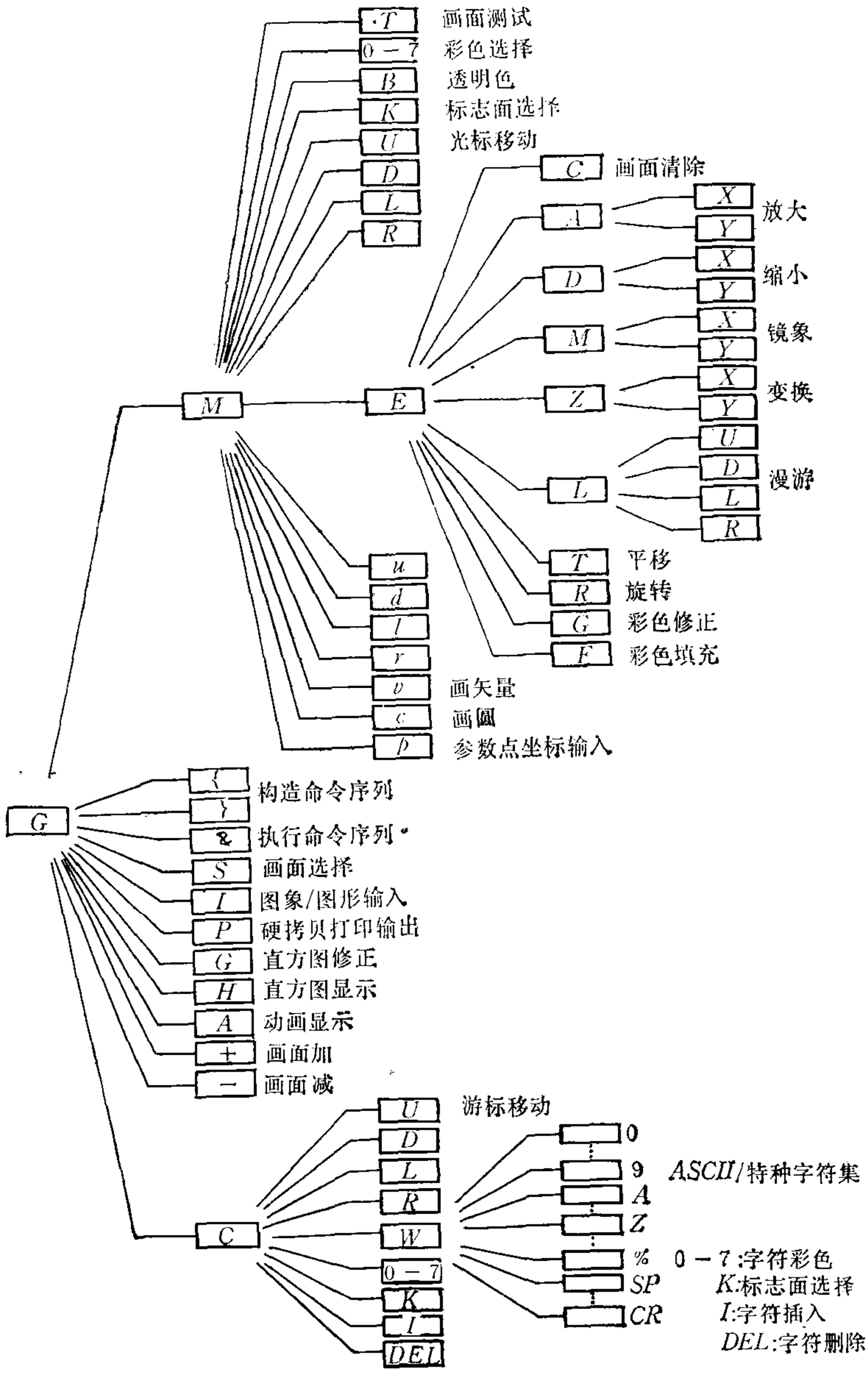

图 3 GCL 命令树的分层结构

令的分层结构。 例如图 3 中，图形（以光标为中心）x 方向放大命令的完整路径名是 **GMEAX.** 如果从命令分层结构的根以下的较低层开始，可以构造部分路径名标识同一命令。本例中，在移动光标选择好图形放大中心后，系统将位于第二层节点 M，这时图形 x 方向放大命令的路径名就是 **EAX.**

与树的结构递归特征相似，GCL 命令树的终端(叶)节点大多具有递归特性，即这些节点自身又可看作是它们所属那棵子树的根。 具体地说，GCL 程序沿指定路径到达某终端节点执行该路径标识的命令后，控制将自动转移到包含该叶节点的子树的根节点上。

因此从第二层节点 M 开始，路径名 **EAXY** 所标识的命令序列将使图形分别在 x，y 方向上放大一倍。

为了在分层路径中来回移动，GCL 提供一种控制转移符〈ESC〉. 它中止节点的递归特性，使控制"上"移一层，因此可以将控制符〈ESC〉想象为一座桥，它由分枝节点通向同一层中的其它节点。例如路径 **EAXY**〈ESC〉**DXY** 可将先前在 x，y 方向上放大的图形重新复原(缩小)。

GCL 的扩充方便灵活，添加新的功能相当于让命令树发芽，长出新的子树. 由于不同层或同一层但不属于同一子树的节点均可重名，这使得 GCL 能利用统一的键盘操作实现图形、字符等多种命令模式。

3.2. 交互工具

对不同的图形应用程序的分析表明，下述两种基本的图形交互操作几乎能概括所有的图形输入要求[6]:

1）在画面任意位置输入一个不依赖任何显示图形的点；

2）标识某个显示图形对象，以便进行图形的后继处理。

GCL 提供屏幕光标与图形窗口这两种交互工具实现上述操作。

光标在命令树第二层节点 M 处产生。此时用彩色代码和 U/u, D/d, L/l, R/r 的集合组织路径 (大小写字母分别表示位移量为 16/1 个象素单位)，光标即可在画面上移动，画出不同颜色的点和线. 光标的另一交互功能是指定参数点坐标，这使得用 GCL 描述某些参数图形(圆与矢量)十分简便。例如光标移动至画面某定点 p_1 后，命令 **P** 记入该点坐标，命令 **Rl** 使光标右移 15 单位至圆周上某点 p_2，因此整体命令序列 **PRIKC** 将在标志面上画出以 p_1 为圆心，半径 $p_1p_2 = 15$ 的圆。如果要在上述封闭域中填充 $5^{\#}$ 色，可将光标移至圆内任一点，使用命令 **EF**5.

GCL 利用图形窗口标识某个显示图形对象，操作者可在所选画面上增加或删除一个(或一组)图形。窗口在每幅画面的标志面上产生，其大小和位置可以任意设定。例如命令序列 GMRDKRRDLLU 定义了一个 32 × 16 (宽 × 高)的窗口，其基准点(左上角)坐标位于 (16, 16).

配合画面选择与光标定位，GCL 提供一组命令使窗口标识的图形能够在所选画面上或画面间变换移动。图 4 给出了一个多画面图形编辑的例子。例中设画面 A, B 已各有一个用窗口标识的图形。沿分层路径 **GS**B**M** 进入系统，选择 B 为前台显示画面，用后继命令即可实现图中所示的编辑操作:

光标位置	GCL 命令	执　行　结　果
—	**GS**B**M**	选择 B 为显示画面，产生光标.
p_1	**ET**A	画面 A 窗口标识的图形平移复制至显示画面 B 的 p_1 处.
p_2	**EMY**B	在 p_2 处产生画面 B 窗口标识图形的 y 轴镜象图形.
p_3	**EMX**A	将画面 A 窗口标识图形的 x 轴镜象图形移至画面 B 的 p_3 处.
p_4	**ER**90A	画面 A 窗口标识的图形平移至画面 B 的 p_4 处，旋转 90°.
p_5	**ER**60 B	画面 B 窗口标识的图形平移至 p_5 处，旋转 60°.

用 GCL 编制程序如同在命令的分层路径中来回行走，程序设计问题被转化成一串由当前位置选择后继路径的问题。为此，系统在显示屏幕左上角的字符显示窗中随时给出路径提示作为用户的向导，它使用户在每一程序步之后都能及时了解自己的准确位置，从而正确地选择后继路径。即使是很少有或者没有程序知识的用户，也能很快掌握它的使用方法。

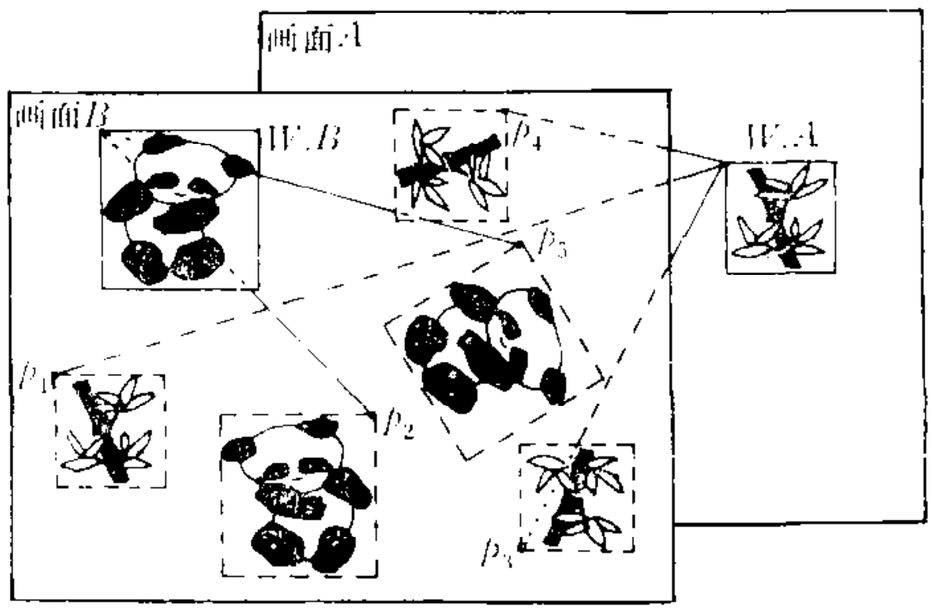

图 4 多画面图形编辑

3.3. 执行方式

GCL 有两种执行方式：交互命令单步（step-by-step）方式与程序（Sequence）方式。前者通常用于程序编制，系统记忆由命令“{”开始，至命令“}”结束的全部路径构成的命令序列，并将它作为可执行文件记入软盘。以上过程可用键盘命令的单步操作实现，其中每一程序步的执行结果均能在屏幕上显示，并允许随时对显示图形进行修改。因此以这种执行方式编制程序，其编制、调试与执行过程是同时完成的。

GCL 可执行文件具有以下格式：

〈程序名〉{〈命令序列〉}.

它通常是一幅或多幅画面生成过程的具体描述。与直接存贮整幅画面相比，描述该画面的程序文件大大压缩了存贮空间。如果需要调入该画面重新显示，可以使用 GCL 的程序执行方式命令：&〈程序名〉. 它能立即在显示屏上动态地生成程序文件描述的图形。

四、结 束 语

计算机图形的应用取决于图形处理系统的成本和灵活性。本文阐述一个基于这一思想而开发的多画面交互式彩色图形系统。它通过一个微计算机集中控制的图象输入/显示装置和多个共享的画面存贮模块，一种容易使用与扩充的图形命令语言，实现了包括图象、图形、字符编辑处理与动画显示在内的多种应用环境所期望的功能，其应用范围可以涉及数字图象处理、统计数据图形显示、自动控制过程监视及计算机辅助设计等方面(图5)。

文中阐述的多画面存贮模块结构，在八位微计算机系统中实现时，因不易获得更高的图形分辨率和处理速度而受限制，但其实现方法同样能够用来构造性能更好的十六位微计算机图形系统。实际应用系统的选择往往应在成本与性能之间进行综合考虑。

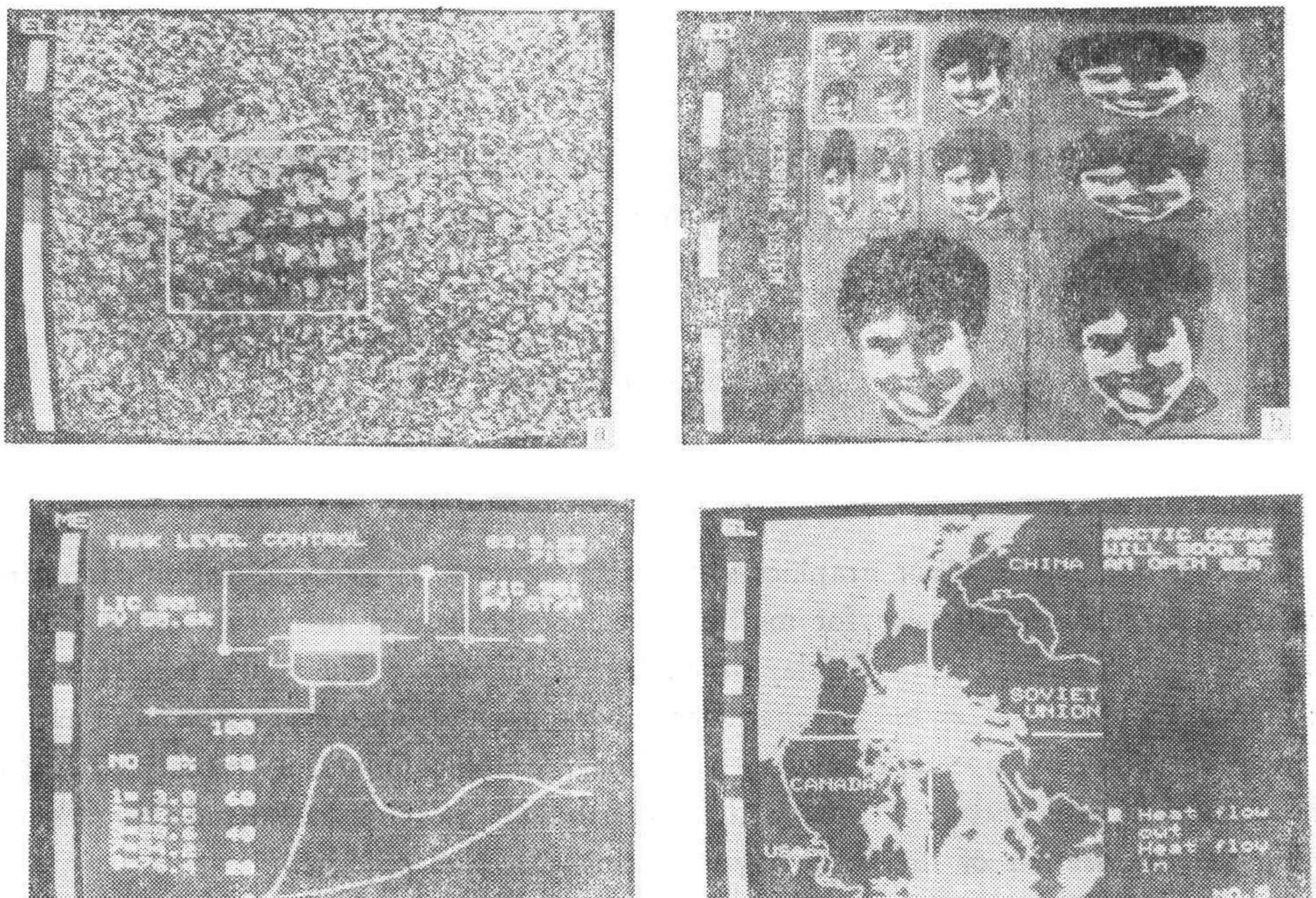

图 5　系统应用实例

本系统显示控制部分的设计是对许鹤群、朱安邦同志工作[7]的修改与扩充，特此致谢.

参　考　文　献

[1] Sutherland, I. E., Ten Unsolved Problems in Computer Graphics, *Datamation,* **12**(1966), 22—27.

[2] Burchi, R. S., Interactive Graphics Today, *IBM System Journal,* **19**(1980), 292—313.

[3] Nisen, W. G. and Franklin, W. R., The Maturation of Computer Graphics, *ICP Interface Manufacturing Engng.,* **3**(1978), 5—11.

[4] Lerner, E. J., The Computer Graphics Revolution, *IEEE Spectrum,* **18**(1981).

[5] 郑君兰，一种低成本的视频数字图象获取方法在微计算机系统中的实现，自动化学报，**11**(1985)，291—299.

[6] Bleher, J. H., Caspers, P. G., Henn, H. H. and Maerker, K., A Graphic Interactive Application Monitor, *IBM System Journal,* **19**(1980).

[7] 许鹤群，朱安邦，TV 伪彩色图象显示终端，信息与控制，**13**(1984).

A MICROCOMPUTER BASED MULTIPICTURE INTERACTIVE COLOUR GRAPHICS SYSTEM

ZHENG JUNLAN

(China Textile University)

ABSTRACT

A low-cost interactive colour graphics system that supports various application environments is described. Hardware assistance for multipicture management is provided, using a method that allows foreground-background display processing of pictures. The system has four display modes: colour graphics and/or colour character modes, pseudocolour image mode and animation mode. Most of the interactive functions, such as image/graphics inputting, modifying and editing, are defined by the system software and an expandable graphic command language, thus permitting the exploration of different ways of using this kind of system.