

图形压缩编码和解码在 自动控制系统中的应用

徐鸿章 李秀琴

(郑州大学) (数理力学系)

提 要: 在自动控制中,为了解决存贮图形占内存空间大的困难,作者编制了一套图形压缩编码和解码,压缩数倍一般可达20倍左右,提高了内存空间的利用率。计算机进行分析处理时,可以用解码恢复原来图形。

关键词: 压缩编码 解码 利用率 像素

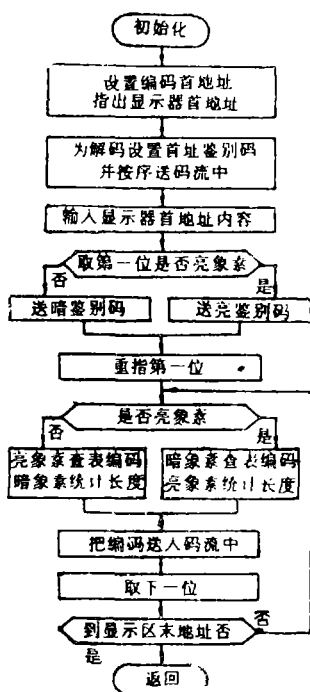
1 概 述

在系统自动控制中,用图象来描述系统控制过程是较理想的方法。为了以后分析对比,图象的存贮也是很重要的。一般存贮图象有两种办法,一种是当形成整幅图形后用打印机把它打印下来,这仍是一种有效的办法,然而缺点是要增加打印设备且图象不能重返屏幕,从而计算机也就不能再对原图象作分析和处理。另一种是把图形存贮在RAM中,或磁盘中,以备复查处理时重新显示出图形来。用存贮的办法必须考虑有大容量的内存空间,磁盘容量虽大,但磁盘系统要额外增加系统的设备费用和复杂性。像Apple—II机上用的显示器,一幅图形就要占内存6k字节,而对过程控制的计算机来说,往往内存资源有限,不能满足同时贮存几幅图形的要求。如果能采取措施把图形压缩得很小,使得同样的内存空间可十倍或几十倍甚至百倍于未压缩之前的利用率,这将是一个创新。本文介绍的图形压缩编码和解码就能达到这样的目的,当需要复查时,用解码来恢复原来的图形。

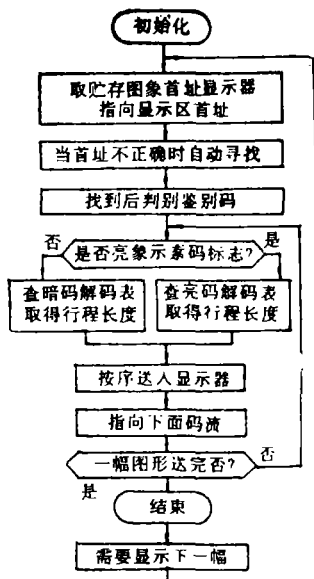
2 图形压缩编码和解码

每幅图形在高密度显示方式下,都是由亮暗像素组成的。如前所述在Apple—II机上用的显示器,图象要占显示区6k字节 共有49152个bit,一个bit组成一个像素,以统计亮暗像素的行程来代替逐个像素的按位显示。这是解决压缩的第一个问题。另一个问题是一幅图形中亮暗行程的分布是不均匀的,一般的曲线图亮程出现的机率以1bit的为多,因此编码时用短码表示,而暗区较大,暗程大都在6bit以上,因此用较长码表示。为了考虑特殊情况,例如一条直线其亮程是很大的,这就要用长码来解决。这样短码中是由短码头和长码尾组成,较长码也由短码头和长码尾组成,不同的仅仅是截断的bit数不一样而已,对行程短的情况码尾可以不用。精选码头和码尾的长度使它和亮暗程相匹配,就能达到高度压缩的目的,根据以上的想法,成功地编写出编码程序如图1流程图所示的和解码程序如图2流程图

所示,前者可将图形一幅接一幅经过压缩送入内存,后者则相反过程,可以一幅接一幅取出来恢复原来图形,并且有自动寻址的功能。两个程序共占内存486字节,程序采用Z80汇编语言。



1图



2图

3 两个实例

压缩编码所占内存的多少是随着图形复杂程度而有所增加的,这可用压缩倍数 R 来表示,

$$R = \frac{W}{B}$$

W 表示压缩前的字节数, B 为压缩后的字节数, R 越大,内存空间的利用率越高。

一台微型单板机在自控过程中,存贮图形的困难很大,因为内存资源有限,贮存两幅图形就要占内存12k字节,这在一般情况下是办不到的事。采用压缩编码后6k字节就能贮存十几幅图形。下面有两幅图形作为例子。

例1:图3是单通道A/D转换,模拟量采用双半波整流,经不断采样变换在屏幕描制图形,当描完一幅图形后通过压缩编码贮存在2000H开始的内存单元中,其内容如图5所示。占内存448字节, $R = 6144 / 448 = 13.7$,压缩近14倍,图4为用解码恢复的图形,和图3对比是完全一样的。

例2:图6为8路A/D转换图形,分别用可调电位器分压的方法作模拟量,形成一幅图形后经过压缩编码存在2000H开始的单元中,其内容如图8所示,占内存384字节, $R = 6144 / 384 = 16$,压缩16倍,图7为采用解码恢复的图形,可以看出和图6也是一样的。图形解码时

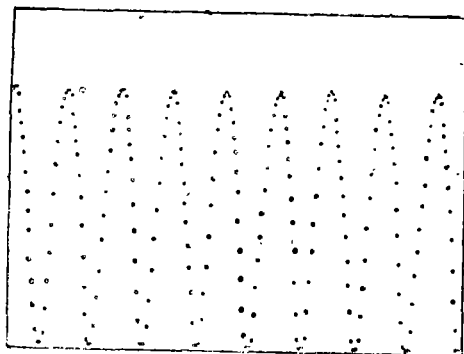


图 3

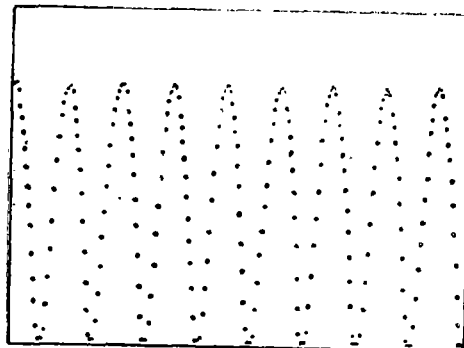


图 4

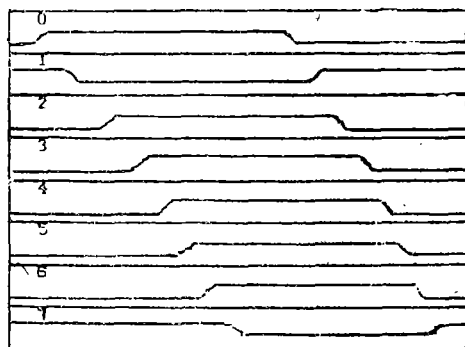


图 6

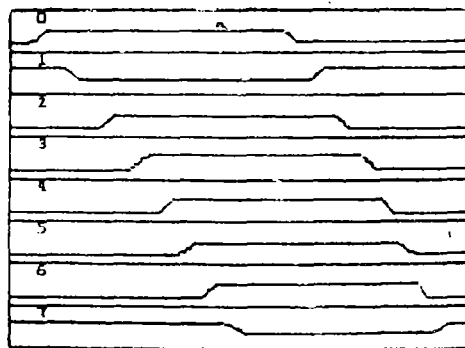


图 7

```

2000  00 00 00 00 01 00 05 96
2008  70 77 02 44 0E 88 0D C0
2010  E1 01 D8 1C 81 C8 1E 81
2018  08 1E 20 37 03 93 07 30
2020  37 03 F0 37 02 30 ED 50
2028  3B 10 D0 EB 03 F0 35 03
2030  90 2D 03 B0 33 70 31 0A
2038  F0 43 03 B0 00 02 59 0A
2040  F0 B1 11 B0 6D 14 D1 61
2048  0D 70 AF 19 51 43 16 10
2050  00 03 4D 03 BB 0D 70 00
2058  02 3B 02 D0 F7 03 B0 D9
2060  02 B1 35 1F 10 E5 03 B1
2068  3F 12 50 3B 19 90 D7 11
2070  70 49 0D 90 9F 0C 10 3B
2078  08 B0 11 00 00 23 90 00
2080  03 15 0D 70 4D 00 00 43
2088  B0 D7 14 D0 13 0D 91 61

```

```

2090  08 90 4F 09 B0 75 04 F0
2098  000 2 D7 00 00 23 B1 73
20A0  07 50 00 02 EB 16 50 D7
20A8  03 B1 25 13 71 03 0D 91
20B0  61 00 00 2D 70 00 02 50
20B8  03 90 D9 00 00 2C 90 97
20C0  00 00 2D 70 CB 03 90 00
20C8  03 90 0D 11 69 19 10 00
20D0  02 1F 0B 91 61 0B B0 1B
20D8  00 00 65 91 E1 00 00 27
20E0  50 F5 0E B0 3B 07 F1 27
20E8  0F 50 00 04 1D 17 D0 F7
20F0  00 00 21 B0 CF 0D 91 91
20F8  00 00 4B 70 75 05 B0 00
2100  02 9D 00 00 45 30 97 00
2108  00 23 B0 00 02 17 17 11
2110  03 1C 50 00 02 9D 02 10
2118  00 03 61 0D 91 89 16 10

```

```

2120 75 0E F0 39 00 00 38 90
2128 00 03 C3 00 00 46 10 FF
2130 13 90 3B 00 00 21 30 00
2138 06 D9 16 11 87 06 30 D7
2140 16 10 63 07 50 00 04 D4
2148 11 30 00 02 D9 1D 51 25
2150 09 D0 00 04 9F 00 00 44
2158 B1 EF 00 00 23 B0 AD 16
2160 10 8D 0F 90 00 03 3F 16
2168 10 00 02 D9 03 BD 00 00
2170 51 71 D1 09 D0 00 02 3B
2178 00 00 66 71 15 13 10 2F

```

```

2180 1C 30 E5 16 11 87 00 00
2188 34 31 61 11 F0 00 03 B9
2190 03 B0 00 03 25 1C D0 9D
2198 03 B0 00 05 81 0E 90 00
21A0 02 43 0D 91 1D 03 B7 00
21A8 00 38 71 47 16 11 1B 14
21B0 86 06 8C 0D 20 1A 30 34
21B8 60 6C C0 D9 81 A3 03 18
21C0 00 00 00 00 00 00 00 00
21C8 00 00 00 00 00 00 00 00
21D0 00 00 00 00 00 00 00 00
21D8 00 00 00 00 00 00 00 00

```

图 5

```

2000 00 00 00 00 02 00 00 80
2003 01 13 1F 97 1F 77 1F 77
2010 1F 77 1F 77 1F 86 00 01
2018 40 42 14 3C 90 85 87 78
2020 88 20 E9 11 70 E5 11 B0
2028 E1 11 F0 D6 82 44 1A C0
2030 00 27 A8 00 04 00 09 47
2038 F1 0F F8 FF 8F F8 FF 8F
2040 E3 00 00 3D 80 3E 24 01
2048 44 0F 90 8D 87 22 11 50
2050 EB 11 10 F 10 D0 F3 10
2058 90 F7 10 50 FA 10 40 00
2060 1D 5C 00 02 00 04 4C 7E
2068 5C 7F C7 F4 7F 47 F4 7F
2070 28 00 05 26 07 B1 09 0F
2078 71 02 41 F6 1F B2 00 83
2080 E4 7E 90 84 87 88 87 84
2088 40 33 11 20 88 00 01 00
2090 00 00 20 00 44 C7 E5 C7
2098 FC 7E 90 00 01 00 8F BB
20A0 8F C3 00 00 67 40 EE 22
20A8 21 DE 21 48 7E C3 DC 41
20B0 43 E9 07 F8 80 87 C2 1F
20B8 B1 08 40 00 04 00 1E 40
20C0 00 25 D8 00 04 00 09 C7
20C8 F1 0F D9 8F CA 8F C5 1F
20D0 D1 FC 60 00 11 18 1D 44
20D8 54 7F 43 B4 43 C3 C4 42
20E0 C3 D4 41 C3 E4 40 C0 00

```

```

20E8 0B F4 15 82 A8 80 02 B0
20F0 00 08 00 00 01 00 02 B5
20F8 3E E3 FD 00 00 08 04 70
2100 C7 DD C7 E1 80 00 55 30
2108 72 11 B0 E5 11 70 E8 48
2110 1C 83 BC 42 C3 D4 40 9F
2118 7C 30 00 02 00 0F C0 00
2120 12 84 00 02 00 04 C4 7E
2128 5C 7D C7 FA 0F CB 8F BB
2130 3F C3 00 00 8B E0 E2 23
2138 A3 FA 3F A1 CE 22 48 00
2140 00 BA C4 3C 3C 44 2C 3D
2148 40 C0 36 07 C0 18 00 00
2150 80 00 00 10 00 20 A3 EE
2158 E3 FA 3F A3 FE 3F E3 FE
2160 00 00 7B 41 E0 00 00 80
2168 20 76 00 E0 F8 41 C8 84
2170 61 26 F8 90 20 D9 12 70
2178 D0 92 54 1A 00 00 40 F1
2780 00 00 00 00 00 00 00 00
2188 00 00 00 00 00 00 00 00
2190 00 00 00 00 00 00 00 00
2198 00 00 00 00 00 00 00 00
21A0 00 00 00 00 00 00 00 00
21A8 00 00 00 00 00 00 00 00
21B0 00 00 00 00 00 00 00 00
21B8 00 00 00 00 00 00 00 00
21C0 00 00 00 00 00 00 00 00
21C8 00 00 00 00 00 00 00 00

```

图 8

间约3秒钟。

从原理上来说,压缩编码和解码程序可推广应用到其它图形的压缩,对屏幕汉字图形的压缩也同样有效。应用于搞自控的计算机中,虽然其内存空间有限也同样可以实现图形的存贮。

图形压缩编码和解码仅仅是自动控制系统中的一部分,使用时由主程序自动调用,当描完一幅自控系统图形后自动调入内存,并经过压缩编码,图形存贮区始终保持最近的十幅图形,并采用循环记录。主程序中设置有专用按键,按一次按键就可以把记录图取出来,并通过解码恢复原来的图形,再按一次,往下再取一幅图。检查完毕按系统执行键,系统继续运行。图形打印和描制可参考[1]。

参 考 文 献

[1]徐鸿章编写“八路二维点游图形在自动控制系统中的应用”,郑大学学报自然科学版 1988年第二期

The application of figurative compression code and decode to the Automatic Control System

Xu Hongzhang Li Xiuqin

(Zhengzhou Institute of Technology)

Abstract: In autocontrol in order to selve the difficult problem that figurative store occupies too much inte—rnal storable space, the writer designed a set of figurative compression code and decode, of which the ratiou may reach about twenty—fdd•AS aresult the utility of internal storable space was increased. When electronic computer is going to make analysis and processing, the primary figure may be resumed by decoding.

Keywords: Compression code; Decode; utiltiy pixel