

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.

G06F 3/14 (2006.01)

H04L 29/06 (2006.01)



[12] 发明专利说明书

专利号 ZL 200610066942.5

[45] 授权公告日 2008年8月13日

[11] 授权公告号 CN 100410868C

[22] 申请日 2006.3.30

[21] 申请号 200610066942.5

[73] 专利权人 华为技术有限公司

地址 518129 广东省深圳市龙岗区坂田华为总部办公楼

[72] 发明人 谢建斌

[56] 参考文献

CN1130970A 1996.9.11

US5917552 1999.6.29

CN1128099A 1996.7.31

审查员 刘 栩

[74] 专利代理机构 北京德琦知识产权代理有限公司

代理人 宋志强 麻海明

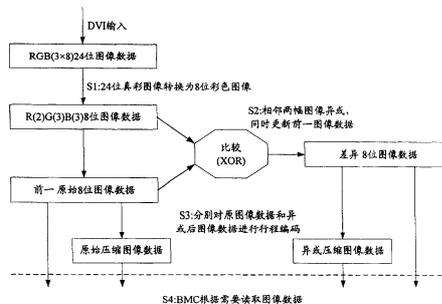
权利要求书 2 页 说明书 7 页 附图 2 页

[54] 发明名称

一种视频数据压缩方法

[57] 摘要

本发明公开了一种针对 KVM 应用的视频数据压缩方法，在该方法中：对于第一帧图像，将该图像数据进行编码压缩，得到图像编码数据，然后将该图像数据和图像编码数据当中数据长度较小的数据作为结果输出；对于第一帧之后的图像，将当前帧图像数据与前一帧图像数据相比较，获取两者的差异数据，并分别对当前帧图像数据和差异数据进行编码压缩得到当前帧图像编码数据和编码的差异数据，然后输出当前帧图像数据、当前帧图像编码数据、差异数据以及编码的差异数据当中数据长度最小的数据。 本发明采用算法简单，便于以硬件实现，极大节省了硬件成本和 PCB 的空间，降低了 KVM 功能实现的成本，同时也提高了 KVM 图像传输的实时性。



1、一种视频数据压缩方法，其特征在于，该方法包括：

对于第一帧图像，执行如下处理：

A1. 将第一帧图像数据进行编码压缩，得到第一帧图像编码数据；

A2. 将第一帧图像数据和第一帧图像编码数据当中数据长度较小的数据作为结果输出；

对于第一帧之后的图像，执行如下处理：

B1. 将当前帧图像数据与前一帧图像数据相比较，获取两者的差异数据；

B2. 分别对当前帧图像数据和差异数据进行编码压缩，得到当前帧图像编码数据和编码的差异数据；

B3. 将当前帧图像数据、当前帧图像编码数据、差异数据以及编码的差异数据当中数据长度最小的数据作为结果输出。

2、根据权利要求1所述的方法，其特征在于，

步骤 B1 中所述比较为将当前帧图像数据与前一帧图像数据相减，所述差异数据为将当前帧图像数据与前一帧图像数据相减得到的数据；或者

所述比较为将当前帧图像数据与前一帧图像数据做异或运算，所述差异数据为将当前帧图像数据与前一帧图像数据做异或运算得到的数据。

3、根据权利要求1所述的方法，其特征在于，

所述图像数据为第一格式的图像数据；

所述步骤 A1 和/或步骤 B1 之前进一步包括：将从视频接口获取的图像数据转换为第一格式的图像数据。

4、根据权利要求3所述的方法，其特征在于，所述第一格式的图像数据为8比特彩色图像数据，所述从视频接口获取的图像数据为24比特真彩图像数据。

5、根据权利要求1所述的方法，其特征在于，步骤 A1 和/或步骤 B2

中所述的编码压缩为行程编码。

6、根据权利要求 5 所述的方法，其特征在于，所述编码压缩的数据编码方式为：用 1 比特表示是否压缩；用 M 比特表示数据的长度；用 N 比特表示数据，其中 M 和 N 分别为正整数。

7、根据权利要求 6 所述的方法，其特征在于，所述表示数据长度的 M 比特分为 k 部分，

当数据长度用二进制表示的位数小于等于所述 k 部分中第一部分的比特数时，用第一部分表示数据长度；

当数据长度用二进制表示的位数大于所述 k 部分中前 L 部分的比特之和且小于等于前 L+1 部分的比特数之和时，用前 L+1 部分表示数据长度，其中 k、L 为正整数，且 L 小于 k。

一种视频数据压缩方法

技术领域

本发明涉及图像压缩技术领域，特别是一种用于多计算机切换器（KVM）的视频数据压缩方法。

背景技术

众所周知，多计算机切换器（Keyboard Video Mouse, KVM）是一种用于服务器远端管理的技术，在 KVM 技术中，服务器的输入/输出设备（多计算机切换器）被放在远端。在使用过程中，将键盘、视频和鼠标的信息进行一定的处理后，通过网络传输到一台或多台服务器，实现从远端对于这些服务器的完全控制和管理。

图 1 为 KVM 的一种实用模式。图中的远端控制台（Management Station）通过局域网集线器（LAN Hub）与服务器主板（Server Main Board）相连接。在服务器端，通用串行总线接口（USB I/F）将键盘和鼠标数据传输给 KVM 模块，视频图像阵列接口（VGA I/F）将视频数据传输给 KVM 模块，KVM 模块将这些键盘、视频和鼠标的的数据实时传输到远端控制台。在传输过程中，需要由 KVM 模块和通过智能平台管理总线（IPMB）与 KVM 模块连接的底板管理控制器（BMC）对视频图像数据进行压缩处理，而控制台会将接收的压缩图像数据解压缩后显示。

对于 KVM 视频图像数据，可以采用视频高密光盘（VCD）的数据处理方式进行压缩。但是该技术的算法比较复杂，需要专用芯片，并且占用较大存储空间，所以成本比较高。并且 VCD 技术为有损压缩，会损失原图的细节。因此 VCD 技术无法与服务器管理领域技术融合，从而无法应用到 KVM 领域。

另外，还可以采用联合图像专家组（JPEG）图像压缩算法对 KVM 视频数据进行压缩。JPEG 算法是专用于图像处理的压缩算法，图像压缩率很高。但是，这种压缩算法复杂，逻辑实现成本较高。并且 JPEG 算法为有损压缩，会损失原图的细节。另外，由于 JPEG 压缩算法复杂，所以压缩时间较长，图像的实时性相对较差。

发明内容

本发明提出了一种针对 KVM 应用的视频数据压缩方法，用以降低 KVM 功能实现的成本，提高 KVM 的图像传输实时性。

根据上述目的，本发明提供了一种视频数据压缩方法，该方法包括：对于第一帧图像，执行如下处理：A1. 将第一帧图像数据进行编码压缩，得到第一帧图像编码数据；A2. 将第一帧图像数据和第一帧图像编码数据当中数据长度较小的数据作为结果输出；对于第一帧之后的图像，执行如下处理：B1. 将当前帧图像数据与前一帧图像数据相比较，获取两者的差异数据；B2. 分别对当前帧图像数据和差异数据进行编码压缩，得到当前帧图像编码数据和编码的差异数据；B3. 将当前帧图像数据、当前帧图像编码数据、差异数据以及编码的差异数据当中数据长度最小的数据作为结果输出。

步骤 B1 中所述比较为将当前帧图像数据与前一帧图像数据相减，所述差异数据为将当前帧图像数据与前一帧图像数据相减得到的数据。或者，所述比较为将当前帧图像数据与前一帧图像数据做异或运算，所述差异数据为将当前帧图像数据与前一帧图像数据做异或运算得到的数据。

所述图像数据为第一格式的图像数据；所述步骤 A1 和/或 B1 之前进一步包括：将从视频接口获取的图像数据转换为第一格式的图像数据。

所述第一格式的图像数据为 8 比特彩色图像数据，所述从视频接口获取的图像数据为 24 比特真彩图像数据。

步骤 A1 和/或步骤 B2 中所述的编码压缩为行程编码。

所述编码压缩的数据编码方式为：用 1 比特表示是否压缩；用 M 比特

表示数据的长度；用 N 比特表示数据，其中 M 和 N 分别为正整数。

所述表示数据长度的 M 比特分为 k 部分，当数据长度用二进制表示的位数小于等于所述 k 部分中第一部分的比特数时，用第一部分表示数据长度；当数据长度用二进制表示的位数大于前 L 部分的比特之和且小于等于前 $L+1$ 部分的比特数之和时，用前 $L+1$ 部分表示数据长度。其中 k 、 L 为正整数，且 L 小于 k 。

从上述方案中可以看出，由于本发明采用算法简单，便于通过硬件实现，极大节省了硬件成本和印刷电路板（PCB）的空间，降低了 KVM 功能实现的成本，并且提高了 KVM 的图像传输实时性。另外，由于算法简单，终端图像数据的恢复也相对简单，便于实现一台终端同时控制多台服务器，从而节省了成本，并提高了效率。

附图说明

图 1 为 KVM 的实用模式图。

图 2 为 KVM 图像处理系统的结构示意图。

图 3 为根据本发明实施例的视频数据压缩方法的流程示意图。

具体实施方式

为使本发明的目的、技术方案和优点更加清楚，以下举实施例对本发明进一步详细说明。

为了便于理解本发明，首先分析服务器管理功能和计算机图像显示特点。在 KVM 应用中，图像大多为简单窗口和菜单界面，因此图像的色彩比较简单，并且图像的变化较少，即使有变化也比较简单，例如窗口的切换、鼠标的移动等。本发明就是针对 KVM 应用中视频图像的特点而提出的一种简单的图像压缩方法。

通常的 KVM 图像处理系统的结构如图 2 所示，包括数字视频接口（DVI）、由现场可编程门阵列（FPGA）实现的图像处理逻辑、由 BMC 实

现的图像传输控制部件以及远端计算机（PC）。

如图 2 所示，DVI 将显示器的模拟视频信号数字化后，转换为红绿蓝（RGB）格式的计算机图像数据格式，输入图像处理逻辑，在逻辑电路中进行图像压缩处理后，由图像传输控制部件将压缩后的图像数据传输到远端计算机，而远端计算机将图像数据恢复后显示出来。

图 3 所示的是根据本发明实施例的图像压缩的流程示意图。参照图 3，该方法包括以下步骤：

步骤 S1：由于在服务器管理的场合，视频图像都是简单的界面，这些界面的色彩比较简单，所以在步骤 S1 中将从 DVI 输入的图像数据格式转换为色彩数据较少的图像数据格式。

如果 DVI 输入的是 24 比特（bit）真彩图像数据，可以将 24bit 真彩图像转换为 8bit 彩色图像。24bit 图像数据中 RGB 每个基色都具有 8bit 数据，这里将其转换为 R 基色取 2bit 数据、G 基色取 3bit 数据、B 基色取 3bit 数据。

步骤 S2：将当前帧图像数据与前一帧图像数据进行比较，比较的方法可以是相减或做异或（XOR）运算，从而得到当前帧图像数据与前一帧图像数据的差异数据。本实施例以异或运算为例说明，那么得到的是当前帧图像数据与前一帧图像数据的异或的图像数据。

在该步骤中，还可以进一步将前一帧图像数据更新为当前帧图像数据，从而为处理下一帧做准备。

步骤 S3：将当前帧图像数据和异或的图像数据分别进行编码压缩。编码压缩的方法有多种，这里可以采用常见的行程编码方法。

步骤 S4：BMC 根据处理后的图像数据长度，控制图像数据传输。具体地，BMC 根据现有的数据：当前帧图像数据、编码后的当前帧图像数据、差异数据、编码后的差异数据，以其中数据长度最小的数据作为结果输出，传输给远程 PC。

需要注意的是，第一次传输数据，由于没有前一帧，所以输出的是当前

帧图像数据和编码后的当前帧图像数据两者中数据长度较小的数据。另外，如果在传输过程中发生传输出错，那么 BMC 也选择输出当前帧图像数据和编码后的当前帧图像数据中数据长度较小的数据。

在上述图像处理过程中，由于同时总共需要存储 4 幅图像，所以对 RAM 只需求同时存储 4 幅图像的存储空间。例如，当图像尺寸为 800×600 时，需要 $800 \times 600 \times 4 = 1920 \text{kByte}$ 的存储空间；当图像尺寸为 1024×768 时，需要 $1024 \times 768 \times 4 = 3146 \text{kByte}$ 的存储空间。

下面具体介绍步骤 S3 中的一种图像数据编码方法，该方法仅采用比较和计数的简单算法，便于逻辑电路的硬件实现。

	Bit31~Bit0	
1	图像数据长度(N+4)	
	Bit15	bit14~bit0
2	是否差值	图像点阵宽度
	Bit15~bit0	
3	图像点阵高度	
	N Byte	
4	图像数据	

表 1 行程编码的图像数据编码

	Bit7	Bit6	Bit5~0
Byte1	是否压缩	长度扩展 1	长度 1
	Bit7	Bit6~bit0	
Byte2	长度扩展 2	长度 2	
	Bit7~bit0		
Byte3	长度 3		
	Bit7~bit0		
Byte4	图像点颜色 R(2)G(3)B(3)		

表 2 长度编码及数据

如表 1 所示，在行程编码的图像数据编码中，数据可以分为 4 个部分，第一部分占 4byte，用来表示图像数据的长度；第二部分占 2byte，其中 1bit 用来表示该数据是否为差异值，例如用 1 表示是和 0 表示否，其余 15bit 用来表示图像点阵宽度；第三部分占 2byte，用来表示图像点阵高度；第四部分占 N byte，为具体图像数据。

表 2 所示为图像数据中的长度编码以及数据。Byte1 中的第一 bit 表示数据是否为压缩数据，例如 1 表示是而 0 表示否。而 byte1 中的其余 7bit 和 byte2、byte3 来表示长度编码。而 byte4 表示这些数据的图像点颜色。

在长度编码中，采用多级的方式实现长度编码。如表 2 所示，如果长度小于等于 63，则

$$\text{长度} = \text{长度 1};$$

如果长度大于 63 而小于等于 8191，则

$$\text{长度} = (\text{长度 1} \times 128) + \text{长度 2};$$

如果长度大于 8191，则

$$\text{长度} = (\text{长度 1} \times 32768) + (\text{长度 2} \times 256) + \text{长度 3}$$

在表 2 所示的长度编码中，最大可表示的长度为 2097151。

换言之，将表示数据长度的 M 比特分为 k 部分，当数据长度用二进制表示的位数小于等于第一部分的比特数时，用第一部分表示数据长度；当数据长度用二进制表示的位数大于前 L 部分的比特之和且小于等于前 L+1 部分的比特数之和时，用前 L+1 部分表示数据长度。其中 M、k、L 为正整数，且 L 小于 k。以表 2 为例，其中 M=23，k=3，那么第一部分为 7bit，第二部分为 8bit，第三部分为 8bit。

在表 2 所示的数据中，在数据为压缩的情况下，即“是否压缩” bit=1，则数据编码格式为：长度编码+1byte 数据；在数据不是压缩的情况下，即“是否压缩” bit=0，则数据编码格式为：长度编码+nbyte 数据。

例如，原始数据为 0x01, 0x01, 0x01, 0x01, 0x01，由于数据相同，所以采取压缩，因此“是否压缩” bit=1。由于数据长度为 5，小于 63，所以

采用长度 1 表示长度，因此表 2 中 byte1 的后 7 个 bit 为 0000101。在此后面加上 1byte 的数据，因此该数据的编码为 0x85, 0x01。

再例如，原始数据为 0x01, 0x02, 0x03, 0x04, 0x05，则编码为 0x05, 0x01, 0x02, 0x03, 0x04, 0x05，其中“0x05”表示不压缩以及长度为 5，“0x01, 0x02, 0x03, 0x04, 0x05”为数据。原始数据为 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x05，则编码为 0x85, 0x01, 0x02, 0x04, 0x05，其中“0x85, 0x01”表示压缩的 5 个“0x01”，长度为 5；“0x02, 0x04, 0x05”表示未压缩的“0x04, 0x05”，长度为 2。

如果，800×600 的一幅纯白图像的编码为 0xce, 0xd3, 0x00, 0xff，那么它的长度为：

$$\text{长度} = ((0xce \& 0x3f) \ll 15) + ((0xd3 \& 0x7f) \ll 8) + 0x00 = 480000 = 800 \times 600$$

以上所述仅为本发明的较佳实施例而已，并不用以限制本发明，凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

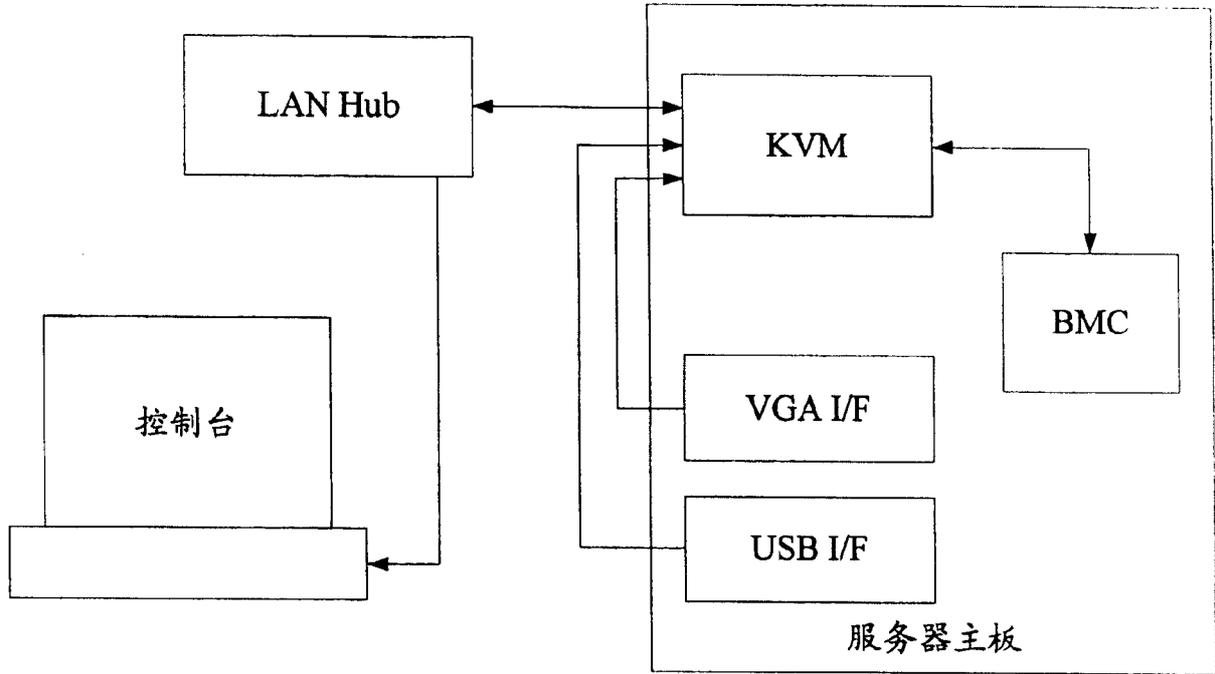


图 1

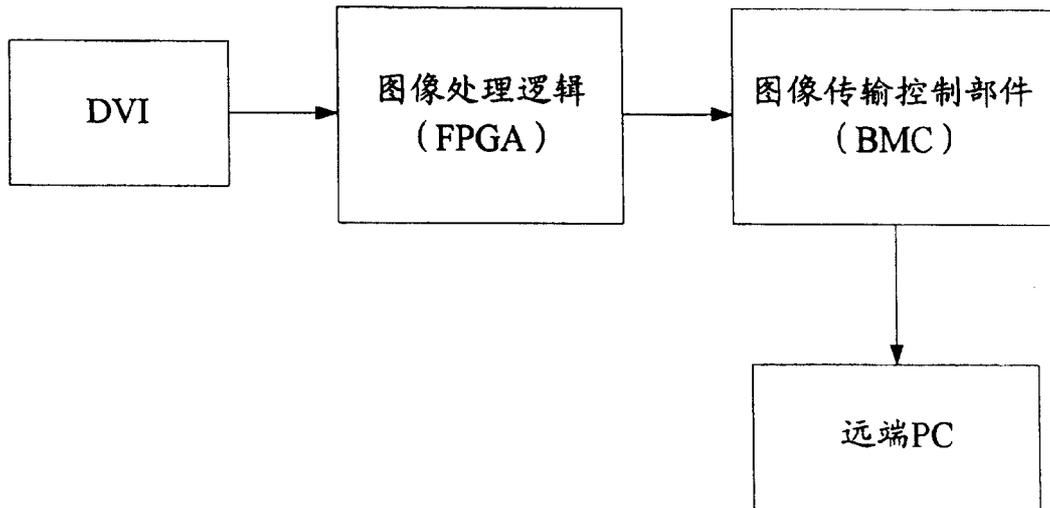


图 2

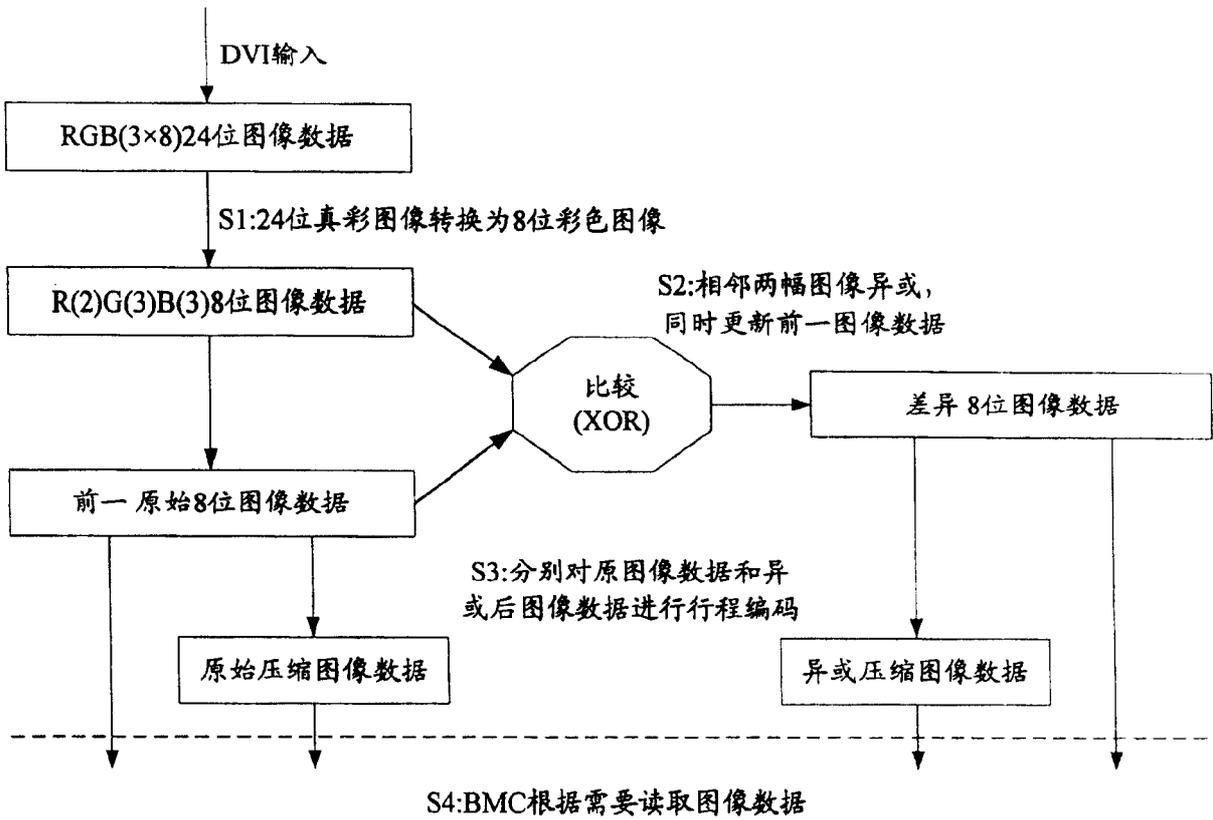


图 3