[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04N 7/24 (2006.01)
H04N 7/26 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200710110193.6

[43] 公开日 2007 年 11 月 7 日

[11] 公开号 CN 101068353A

[22] 申请日 2007.6.18
[21] 申请号 200710110193.6
[30] 优先权
[32] 2006. 6.16 [33] US [31] 60/814,623
[71] 申请人 威盛电子股份有限公司
地址 中国台湾台北县
[72] 发明人 扎伊尔德·荷圣 约翰·柏拉勒斯
徐建明

[74] 专利代理机构 北京市柳沈律师事务所
代理人 葛宝成

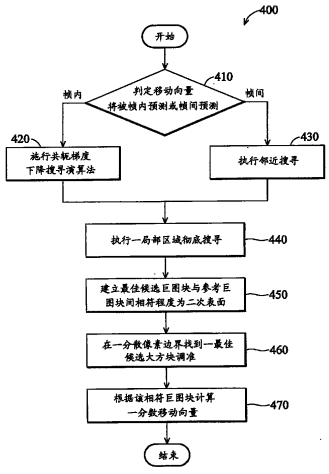
权利要求书 3 页 说明书 19 页 附图 12 页

[54] 发明名称

图形处理单元与计算巨图块的绝对差值加总值的方法

[57] 摘要

一种图形处理单元，包含：一指令解码器，设置成将一绝对差值加总指令解码为多个参数，该多个参数描述在 U、V 坐标上的一 M × N 像素方块与一 n × n 像素方块，其中，M、N、n 是整数；以及一绝对差值加总加速逻辑电路，设置成接收该多个参数并计算多个绝对差值加总值，各绝对差值加总值对应该 n × n 像素方块，及对应该 M × N 像素方块且为该 n × n 像素方块的水平平移的多个方块其中之一。



1. 一种图形处理单元，包含:

一指令解码器，设置成将一绝对差值加总指令解码为多个参数，该多个参数描述在U、V坐标上的一M x N像素方块与一n x n 像素方块;以及

一绝对差值加总加速逻辑电路，设置成接收该多个参数并计算多个绝对差值加总值，各绝对差值加总值对应该n x n像素方块，及对应存在于该M x N像素方块且与该n x n像素方块有一位差。

2. 如权利要求1所述的图形处理单元，其中，该绝对差值加总加速逻辑电路更包含:

多个绝对差值加总计算单元，各绝对差值加总计算单元设置成接收该n x n 像素方块，及接收包含在该M x N像素方块中的该多个方块其中之一，并计算一对应的该多个绝对差值加总值其中之一。

3. 如权利要求1所述的图形处理单元，其中，描述该M x N像素方块的参数定义在一纹理快取中的该M x N像素方块的一地址。

4. 如权利要求1所述的图形处理单元，其中，描述该M x N像素方块的参数定义在一纹理快取中的该M x N像素方块的一基本与一相对地址。

5. 如权利要求1所述的图形处理单元，其中，该位差为一水平位差。

6. 如权利要求1所述的图形处理单元，其中，该M x N像素方块代表一运动估测预测方块，而该n x n像素方块代表一运动估测参考方块 。

7. 如权利要求2所述的图形处理单元，其中，该多个绝对差值加总计算单元可平行地处理数据。

8. 如权利要求2所述的图形处理单元，更包含一第一逻辑电路，将该多个绝对差值加总值累加至一目标缓存器。

9. 如权利要求8所述的图形处理单元，其中，该第一逻辑电路，将该多个绝对差值加总值以一次序结合存入该目标缓存器，该次序由各该M x N中的方块的U坐标而判定。

10. 如权利要求2所述的图形处理单元，更包含:

一纹理高速缓存，设置成将像素数据存储有一预定位数的纹理图像格式;以及

一纹理滤波单元，设置成判断该M x N像素方块是否延伸一纹理图像边

界，并对应自该纹理高速缓存撷取一个或多个在该 M x N 像素方块周围的纹理图像校正 n x n 方块，并结合从该纹理图像校正 n x n 方块按位选择的行与列，使得最左边的位被写入第一滤波缓冲器而最右边的位被写入第二滤波缓冲器。

11. 一种图形处理单元，包含:

一主处理器接口，接收视频加速指令;以及

一视频加速单元，响应该视频加速指令，该视频加速单元包含一绝对差值加总加速逻辑电路，设置成接收该多个参数并计算多个绝对差值加总值，各绝对差值加总值对应该 n x n 像素方块，及对应存在于该 M x N 像素方块且与该 n x n 像素方块有一位差的多个方块其中之一。

12. 如权利要求 11 所述的图形处理单元，其中，该绝对差值加总加速逻辑电路更包含:

多个绝对差值加总计算单元，各绝对差值加总计算单元设置成接收该 n x n 像素方块，及接收包含在该 M x N 像素方块中的该多个方块其中之一，并计算一对应的该多个绝对差值加总值其中之一。

13. 如权利要求 12 所述的图形处理单元，其中，该多个绝对差值加总计算单元可成平行地处理数据。

14. 如权利要求 11 所述的图形处理单元，更包含一第一逻辑电路，将该多个绝对差值加总值累加至一目标缓存器。

15. 如权利要求 14 所述的图形处理单元，其中，该第一逻辑电路，将该多个绝对差值加总值以一次序结合存入该目标缓存器，该次序由各该 M x N 中的方块的 U 坐标而判定。

16. 如权利要求 11 所述的图形处理单元，更包含:

一纹理高速缓存，设置成将像素数据存储有一预定位数的纹理图像格式;以及

一纹理滤波单元，设置成判断该 M x N 像素方块是否延伸一纹理图像边界，并对应自该纹理高速缓存撷取一个或多个在该 M x N 像素方块周围的纹理图像校正 n x n 方块，并结合从该纹理图像校正 n x n 方块按位选择的行与列，使得最左边的位被写入第一滤波缓冲器而最右边的位被写入第二滤波缓冲器。

17. 如权利要求 11 所述的图形处理单元，其中，描述该 M x N 像素方块

的参数定义在一纹理快取中的该 M x N 像素方块的一基本与一相对地址。

18. 如权利要求 11 所述的图形处理单元，其中，描述该 M x N 像素方块的参数直接定义该像素数据。

19. 如权利要求 11 所述的图形处理单元，其中，该 M x N 像素方块代表一运动估测预测方块，而该 n x n 像素方块代表一运动估测参考方块。

20. 一种计算一 M X N 巨图块的一绝对差值加总值的方法，其中，M、N 为整数，该方法包含:

执行一绝对差值加总指令以计算一 M x M 巨图块的一第一 n x n 部分的一第一绝对差值加总值，该第一部分包含该 M x M 巨图块的一左上部分，其中，n 为整数;

执行一绝对差值加总指令以计算该 M x M 巨图块的一第二 n x n 部分的一第二绝对差值加总值，该第二部分包含该 M x M 巨图块的一右上部分;

累加该第一与第二绝对差值加总值得一总和;

执行该绝对差值加总指令以计算该 M x M 巨图块的一第三 n x n 部分的一第三绝对差值加总值，该第三部分包含该 M x M 巨图块的一左下部分;

将该第三绝对差值加总值加至该总和;

执行该绝对差值加总指令以计算该 M x M 巨图块的一第四 n x n 部分的一第四绝对差值加总值，该第四部分包含该 M x M 巨图块的一右下部分;以及

将该第四绝对差值加总值加至该总和。

图形处理单元与计算巨图块的绝对差值加总值的方法

技术领域

本发明涉及一图形处理单元，且特别涉及具有图像压缩与解压缩特征的图形处理单元。

背景技术

个人计算机与消费性电子产品用于各种娱乐用品。这些娱乐用品可以大致区分为 2 类：使用计算机制图（computer-generated graphics）的那些，例如计算机游戏；与使用压缩视频数据流（compressed video stream）的那些，例如预录节目到数字式激光视盘（DVD）上，或由有线电视或卫星业者提供数字节目（digital programming）至一机顶盒（set-top box）。第 2 种亦包含编码模拟视频数据流，例如由一数字录像机（DVR，digital video recorder）所执行。

计算机制图通常由一图形处理单元（GPU，graphic processing unit）产生。一图形处理单元是一种建立在计算机游戏平台（computer game consoles）与一些个人计算机上一种特别的微处理器。一图形处理单元被最佳化为快速执行描绘三度空间基本对象（three-dimensional primitive objects），例如三角形、四边形等。这些基本对象以多个顶点描述，其中，每个顶点具有属性（例如颜色），且可施加纹理（texture）至该基本对象上。描绘的结果是一二度空间像素阵列（two-dimensional array of pixels），显示在一计算机的显示器或监视器上。

视频数据流的编码与解码牵涉到不同种类的运算，例如，离散余弦变换（discrete cosine transform）、运动估测（motion estimation）、运动补偿（motion compensation）、去方块效应滤波器（deblocking filter）。这些计算通常由一般用途中央处理器（CPU）结合特别的硬件逻辑电路，例如特殊应用集成电路（ASIC，application specific integrated circuit），来处理。消费者因而需要多个运算平台以满足他们的娱乐需求。因而需要可以处理计算机制图与视频编码/解码的单一计算平台。

发明内容

本发明的一方面是一种图形处理单元，包含：一指令解码器，设置成将一绝对差值加总指令解码为多个参数，该多个参数描述在U、V坐标上的一M x N像素方块与一n x n像素方块，其中，M、N、n是整数;以及一绝对差值加总加速逻辑电路，设置成接收该多个参数并计算多个绝对差值加总值，各绝对差值加总值对应该n x n像素方块，及对应存在于该M x N像素方块且与该n x n像素方块有一位差。

本发明的另一方面是一种图形处理单元，包含：一主处理器接口，接收视频加速指令;以及一视频加速单元，响应该视频加速指令，该视频加速单元包含一绝对差值加总加速逻辑电路，设置成接收该多个参数并计算多个绝对差值加总值，各绝对差值加总值对应该n x n像素方块，及对应存在于该M x N像素方块且与该n x n像素方块有一位差的多个方块其中之一。

本发明的另一方面是一种计算一M X N巨图块的一绝对差值加总值的方法，其中，M、N为整数，该方法包含：执行一绝对差值加总指令以计算一M x M巨图块的一第一n x n部分的一第一绝对差值加总值，该第一部分包含该M x M巨图块的一左上部分，其中，n为整数;执行一绝对差值加总指令以计算该M x M巨图块的一第二n x n部分的一第二绝对差值加总值，该第二部分包含该M x M巨图块的一右上部分;累加该第一与第二绝对差值加总值得一总和;执行该绝对差值加总指令以计算该M x M巨图块的一第三n x n部分的一第三绝对差值加总值，该第三部分包含该M x M巨图块的一左下部分;将该第三绝对差值加总值加至该总和;执行该绝对差值加总指令以计算该M x M巨图块的一第四n x n部分的一第四绝对差值加总值，该第四部分包含该M x M巨图块的一右下部分;以及将该第四绝对差值加总值加至该总和。

附图说明

图1是用于图形与视频编码及/或解码的一示范性运算平台的方块图。

图2是图1的视频编码器160的功能方块图。

图3A、B说明将目前图像分割成不重迭的区段的巨图块。

图4是图2的运动估测器所使用的算法的一示范性实施例的流程图。

图5是图4共轭梯度步骤440的一实施例的流程图。

图6说明使用图5的共轭梯度下降步骤440的示范状态。

图 7 是图 4 邻近搜寻算法的一实施例的流程图。

图 8A、B 说明图 7 的邻近搜寻算法所使用的 5 个候选巨图块的相对位置。

图 9A、B 是说明对参考与预测方块进行绝对差值加总指令运作的方块图。

图 10 是图 1 的图形处理单元的数据流程图。

图 11 是图 10 纹理滤波单元与纹理快取的方块图。

附图符号说明

100～系统、110～主处理器、120～图形处理器（GPU）、130～存储器、

140～总线、150～视频加速单元（VPU）、160～软件解码器、

170～视频加速驱动器。

205～图像、210～减法器、220～运动估测器、230～参考图像、

245～运动向量、255～预测方块、260～剩余图像、

270～离散余旋转换器、280～量化器、290～熵解码器、2100～解码器。

310～目前巨图块、320～搜寻窗、330～点。

400～程序、410～判定运动向量将被帧间预测或帧内预测、

420～施行共轭梯度下降搜寻算法、430～执行邻近搜寻、

440～执行一局部区域彻底搜寻、

450～建立最佳候选巨图块与参考巨图块间相符程度为二次表面、

460～在一分数像素边界找到一最佳候选巨图块调准、

470～根据该相符巨图块计算一分数运动向量、

505～初始化一候选方块、

510～计算候选巨图块$C_{x,y}$四周的候选巨图块的坐标、

515～分别计算 5 个候选巨图块的绝对差值加总、

520～计算梯度g_x与g_y、525～梯度是否低于一临界值、

530～计算四个新候选巨图块的坐标、

535～对各候选巨图块分别执行共轭梯度下降步骤、

440、540～比较绝对差值加总值是否低于一临界值、

545～回传有最低绝对差值加总值的候选巨图块、

550～选择一新的中央候选巨图块、

555～从梯度g_x与g_y计算新的步骤值Δ_x与Δ_y、

560～测试迭代循环数是否大于一最大值、565～回传不相符、

610C～候选巨图块、610L-610R-610T-610B～四个周围候选、

620X-620Y～初始候选计算梯度、

630TL-630TR-630BL-630BR ～四个新的中央候选巨图块、640L-640R-640T-640B～候选、670-680～候选

710～利用目前巨图块 310 地址的绝对值与每行巨图块数计算一标记变量 TOPVALID、若此绝对值非 0，则 TOPVALID 为真，此外，TOPVALID 为假

720～标记变量 LEFTVALID 是利用目前巨图块地址的除以整数与每行巨图块数计算。若此除数非 0，则 LEFTVALID 为真，此外，LEFTVALID 为假。

730～结合使用 TOPVALID 与 LEFTVALID 变量以判定目前巨图块邻近的 4 个候选巨图块的可得性。

740～为一先前候选巨图块 P 判定可得性。

750～为每一可得候选巨图块计算绝对差值加总。

810-850～候选巨图块。

910-940～4x4 方块、950～4x4 参考方块。

234～旋转逻辑、950～预测方块、960-990～绝对差值加总计算单元、

1010～指令流处理器、1020～指令、1030～指令数据、

1040～执行单元池、1050～纹理滤波单元、1060～纹理快取、

1070～后包装器、1100～视频处理单元。1120～纹理图像、

1130～目标方块、1140-1170～纹理图像、1110A-B～缓冲器。

具体实施方式

在此揭露的实施例提供利用一图形处理单元以增进运动估测系统与方法。

1.用于视频编码的运算平台

图 1 是用于图形与视频编码及/或解码的一示范性运算平台的方块图。系统 100 包含一一般用途 CPU 110（此后称为主处理器）、一图形处理器（GPU）120、存储器 130 与总线 140。图形处理单元 120 包含一视频加速单元（VPU）150，其可加速视频编码及/或解码，将在后叙述。图形处理单元 120 的视频加速功能是可在图形处理单元 120 上执行的指令。

软件解码器 160 与视频加速驱动器 170 位于存储器 130 中，解码器 160 在主处理器 110 上执行。通过一个由视频加速驱动器 170 提供的一接口，解码器 160 亦可发出给图形处理单元 120 的视频加速指令。如此一来，系统 100

通过发出视频加速指令给图形处理单元 120 的主处理器软件（host processor software）执行视频编码。依此法，经常被执行的密集运算方块（computationally intensive blocks）被卸至图形处理单元 120，而更复杂的运算是由主处理器 110 所执行。

图 1 中省略数个对于解释图形处理单元 120 的视频加速特征并非必要且熟悉此项技术者熟知的现有元件。接下来将对视频编码概要说明，再接下来讨论一个视频编码元件（运动估测器）如何利用图形处理单元 120 所提供的视频加速单元功能。

2.视频编码器

图 2 是图 1 的视频编码器 160 的功能方块图。输入至编码器 160 的图像（205）由像素所组成。编码器 160 利用图像 205 内的时间（temporal）与空间相似性(spatial similarities)运作，并且利用判定一帧内（空间）及/或帧间（时间）的差异相似性编码。空间编码利用一图像内邻近像素通常相同或相关的特性编码，故仅对差异编码。时间编码利用一连串图像中的许多像素通常相同的值，故仅对图像间的差异编码。编码器 160 亦利用熵编码的统计冗余性：一些图像较另一些图样更常发生，故较常发生的以较短的码代表。熵编码的范例包含霍夫曼编码（Huffman coding）、运行长度编码（run-length encoding）、算术编码（Arithmetic coding）与前后自我适应的二位算术编码（context-adaptive binary arithmetic coding）。

在此示范性实施例中，输入图像 205 的方块被提供至一减法器 210 与一运动估测器 220。运动估测器 220 比较输入图像 205 内的方块与一预先存储的参考图像 230 以找出相似的方块。运动估测器 220 计算代表相符方块间配置的一组运动向量 245。运动向量 245 与参考图像的相符方块 230 合称为预测方块 255，代表时间编码。

预测方块 255 提供至减法器 210，其将输入图像 205 减去预测方块 255 以产生一剩余图像 260。剩余图像 260 被提供至离散余旋转换器（DCT，discrete cosine transform）方块 270 与量化器 280，其执行空间编码。量化器 280 的输出（例如一组量化后的 DCT 系数）由熵编码器 290 编码。

对于某种类型的图像（信息或 I 帧，与预测或 P 帧），该空间来自量化器 280 的空间编码余数（spatially encoded residual）被提供给内部解码器。解码器利用空间编码余数结合由运动估测器 220 所产生的运动向量 245 以对

空间编码图像 205 解码。重新建构的图像被存储在参考图像缓冲器 295 中，其是提供至运动估测器 220，如前所述。

如结合第一图所讨论的，编码器 160 在主处理器 110 上执行，然而亿利用由图形处理单元 120 所提供的视频加速指令。尤其是，由运动估测器 220 所实现的算法利用由图形处理单元 120 所提供的绝对差值加总（SAD，sum-of-absolute-difference）指令以达成正确的运动估测，在相对低的运算量下。接着将详述运动估测算法。

3. 软件运动估测算法

a. 搜寻窗(Search Window)

如示于图 3，运动估测器 220 将目前图像 205 切割成不重迭的各区段，称为巨图块。巨图块的大小会依编码器所使用的规范（例如，MPEG-2、H.264、VC）与图像的大小而改变。

在此叙述的示范性实施例，与在各种不同编码标准中，一巨图块是 16x16 像素。一巨图块更切割成方块，该方块的大小可为 4x4、8x8、4x8、16x8、或 8x16。

在 MPEG-2 中，各巨图块可仅有一运动向量，故运动估测是根据巨图块。H.264 允许达 32 个运动向量（依程度而定），故在 H.264 中，运动估测是根据 4x4 或 8x8 方块的基础而计算。H.264 的变化，称为 AVS，该运动方块永远为 8x8。在 VC-1 中，其可为 4x4 或 8x8。

运动估测算法 220 对目前图像 205 中的每一巨图块执行运动估测，依照在一预先编码的图像 230（其类似于目前图像 205 的巨图块）中寻找一方块的目标。参考图像 230 中的巨图块与目前图像 205 中的巨图块间的置换是计算并存储为运动向量（245，图 2）。

为方便说明，运动估测程序将以目前图像 205 中一特定巨图块说明（310）。此范例所选择的巨图块 310 是在目前图像 205 的中间，然而相同技术亦应用在其它巨图块。

一搜寻窗（320）在参考图像 230（对应目前图像 205 的巨图块 310）中巨图块的中间。即，若巨图块 310 位于（X，Y），则在参考图像 230 中的搜寻窗 320 亦位于（X，Y），如示于点 330。其它实施例将巨图块放在参考图像 230 的其它部分，例如左上。范例图 3 中的搜寻窗 320 在水平方向延伸通过相应巨图块的两像素，在垂直方向一个像素。因此，搜寻窗 320 包含 14 个不同巨

图块：两个巨图块分别发现1个与2个像素，就在位置330的左边；另一组两个巨图块在位置330的左边；剩下组在位置330的上面、下面、左上、又上、左下与右下。

由运动估测器 220 所执行的相符方块运动运算使用绝对差值加总作为判断巨图块间相似性（相符）的准则。对绝对差值加总，计算两像素值的差值绝对值，并将一方块中所有像素的这些差值绝对值加总，如熟悉该项技艺的人士所理解的。运动估测器 220 结合使用绝对差值加总准则与选择待测相似性的目标巨图块的开创性方法，其将在下面说明。

b.选择目标巨图块

运动估测器 220 使用不同的搜寻方法，依据运动估测器 220 是产生目前图像 205 的内部编码(intra-coded)运动向量或外部编码(inter-coded)运动向量。运动估测器 220 利用真实世界关于运动的现有知识以预测该相符巨图块应该在搜寻窗 320 的何处，减少搜寻窗 320 中目标方块数目，其实际与目前图像 205 中的巨图块 310 进行相似测试。在真实世界中，物体通常以固定加速度运动，这表示我们可以期待一帧（光学流 optical flow）中物体的运动是缓和且相似（即实质上连续）的，在空间上与时间上都是。此外，在绝对差值加总表面（即在一搜寻空间描绘绝对差值加总值）被期待为相对地缓和（即相对少数量的局部最小点）。

利用此现有知识需要指挥搜寻最可能发现最相符的地方，在此揭露的算法使用减少要被执行搜寻的数目以找到较佳的最小点。如此一来，该算法在计算上有效率也可有效的标出较佳的相符。

图 4 是一示范性实施例运动估测器 220 用来计算目前图像 205 内一目前巨图块 310 的运动向量的算法流程图。运动估测程序从步骤 410 开始，其判定由运动估测器 220 为目前图像 205 所产生的运动向量将被帧间预测(inter-predicted)或帧内预测(intra-predicted)。若使用帧内预测则接着进行步骤 420，在此施行共轭梯度下降搜寻算法（conjugated gradient descent search algorithm）以寻找搜寻窗 320 内一预测巨图块，这与参考巨图块（目前图像 205 内的目前巨图块 310）是较佳的相符。共轭梯度下降搜寻算法（步骤 420）将结合第 5、6 图详细说明。

回到步骤 410，若使用帧间预测以产生运动向量，则接着执行步骤 430，在此执行“邻近的”或“邻近区域”搜寻。该搜寻包含邻近于目前图像 205

内目前巨图块 310 的巨图块，以及对应的先前编码参考图像 230 内的巨图块。邻近搜寻算法（步骤 430）将结合第 7、8 图详细说明。

共轭梯度下降搜寻算法（步骤 410）与邻近搜寻算法（步骤 430）各从一大群目标预测巨图块中认出了较佳或可接受的相符。熟悉此项技艺的人士应当了解到用来判定如何才是一个“较佳的相符”的准则可以是相对的或是绝对的。例如，在此叙述的邻近搜寻算法使用一绝对准则：有最低值（score）的目标巨图块被视为较佳的相符。然而，在此叙述的共轭梯度下降搜寻算法利用一临界值，绝对差值加总值低于该临界值的第一方块被视为较佳的相符。然而，该临界值的准则是一设计或实现决定。

在处理步骤 420 或 430 之后，以认出一较佳候选相符。步骤 440 更执行一局部区域彻底搜寻（local area exhaustive search）以找到最佳的候选。该搜寻区域位于步骤 420 或 430 所认出的较佳候选巨图块附近。在一些实施例中，在执行步骤 420，共轭梯度下降搜寻算法之后（即在帧内预测的状况下），局部彻底搜寻所搜寻的区域包含步骤 420 所认出的局部最小值（较佳候选）的外面附近的 4 个对角。例如，若在梯度下降上个步骤所使用的值是 1，则该搜寻限制在离该较佳候选（±1,±1）的点。在一些实施例中，当执行步骤 430 之后（即在帧间预测的状况下），局部彻底搜寻（步骤 440）所搜寻的包含在较佳候选巨图块附近一小区域的候选，通常是（±2,±2）。

步骤 440 的局部彻底搜寻从一较佳候选巨图块限缩至一最佳候选巨图块，这是像素调准（pixel-aligned），即具有整数像素分辨率。步骤 450 与 460 在一分数像素边界（fractional-pixel boundary）找到一最佳候选巨图块调准。现有分数运动搜寻算法使用特定编解码器滤波算法（codec-specific filtering algorithm）以内插在分数位置的像素值，根据周围的整数位置。相对的，步骤 450 建立最佳候选巨图块与参考巨图块间相符程度为二次表面，而步骤 460 分析地判定该表面的最小值。最小值对应一最佳相符巨图块，为分数而非整数分辨率。（开创性的以分数分辨率判定最佳相符巨图块的建立模型方法将在后面的段落加以说明。）在有着分数分辨率的相符巨图块在步骤 450 被认出之后，接着处理步骤 470，根据该相符巨图块计算一分数运动向量，使用熟悉此项技艺者所知悉的技术。接着就完成了程序 400。

熟悉此项技艺者应当了解到上面的算法在本质上是连续的，因其使用了邻近区域的信息。尽管使用了硬件加速的现有设计通常避免连续算法，因为

许多原因，连续的设计在这里是适当的。首先，像素数据是以连续水平扫描线的形式（sequential raster fashion）读取，因而可被预先接收，维持在一电路缓冲器中。其次，在含有单一绝对差值加总加速单元的实施例中，效能是限制在该单元是否能维持满载而非连续处理。绝对差值加总加速单元在预测方块没有许多快取遗漏下可以维持高负载。因为遗漏率是快取大小的函式，而 HDTV 分辨率图像在快取中仅需要 1920/8 = <1KB 运动向量，低的快取遗漏率是可以预期的。

c. 使用共轭梯度下降的帧内预测运动向量

图 5 是图 4 共轭梯度步骤 440 的流程图，由运动估测器 220 的一实施例所执行。如前所述，步骤 440 是在判定使用帧内预测将被用来寻找搜寻窗 320 内巨图块是与目前方块 310 为一较佳（即可接受的）相符时执行。绝对差值加总值为了一组 5 个初始候选而计算：目前巨图块、与目前巨图块上、下、左、右的巨图块。从这初始组 5 个绝对差值加总值，计算两组互相垂直的梯度。从这两组梯度，得到最陡峭的方向的梯度。若该梯度相对地浅，或 5 个初始候选巨图块有非常接近的绝对差值加总值，则该搜寻延伸远离目前巨图块，因为在此区域内不存在有较佳局部最小机率的条件的候选。在对共轭梯度下降步骤 440 概述之后，该步骤将更详细的说明于下。

该步骤从步骤 505 开始，在此初始化一候选方块$C_{x,y}$与步骤值Δ_x与Δ_y。在一实施例中，候选巨图块$C_{x,y}$设为搜寻窗 320 的左上角，而步骤值Δ_x与Δ_y均设为一小整数值，例如 8。接着在步骤 510，计算候选巨图块$C_{x,y}$四周的候选巨图块的坐标。这四个候选巨图块是候选巨图块$C_{x,y}$的上、下、左、右四个。即，

$$T=(C_x,-\Delta_y+C_y);R=(\Delta_x+C_x,C_y);B=(C_x,\Delta_y+C_y);L=(-\Delta_x+C_x,C_y)$$

接着处理步骤 515，在此分别计算 5 个候选巨图块的绝对差值加总（原本那个与周遭四个）。在步骤 520，计算梯度g_x与g_y。梯度g_x是左边与右边巨图块绝对差值加总值的差。梯度g_y是上面与下面巨图块绝对差值加总值的差。如此一来，不论可能相符巨图块间的误差值是增加或减少，该梯度表示 x 或 y 方向。在步骤 525，该梯度与一临界值作比较。若该梯度低于该临界值（即该梯度相对地浅），这表示在目前搜寻区域中无局部最小值，故该搜寻延伸至新的候选巨图块。这些新的候选巨图块远离了原本的候选处理巨图块$C_{x,y}$。在一些实施例中，当在步骤 515 为候选巨图块所计算的绝对差值加总值相似时

亦延伸该搜寻。该延伸搜寻继续在步骤 530 进行，在此计算四个新候选巨图块的坐标。原本四个候选巨图块是在$C_{x,y}$上下左右距离(Δ_x,Δ_y)的地方，选择四个新候选巨图块以形成原本候选巨图块$C_{x,y}$周围正方形角落，距离(Δ_x,Δ_y)：

$$TL=(-\Delta_x+C_x,-\Delta_y+C_y);TR=(\Delta_x+C_x,\Delta_y+C_y);BL=(-\Delta_x+C_x,C_y);BR=(\Delta_x+C_x,\Delta_y+C_y)$$

在步骤 535，对这些新的候选巨图块（C, TL, TR, BL, BR）分别执行共轭梯度下降步骤 440。

回到步骤 525 的梯度比较，若在巨图块 520 所计算的梯度等于或大于该临界值（即该梯度相对地陡峭），接着在步骤 540 在步骤 515 所计算的绝对差值加总值与一临界值作比较。若该绝对差值加总值低于该临界值，则表示找到较佳相符，则步骤 440 回到呼叫器（在步骤 545），提供该呼叫器有最低绝对差值加总值的候选巨图块。

若在步骤 540 所测试的该绝对差值加总值等于或低于该临界值，表示没有找到较佳相符，故调整搜寻区域。在步骤 550，选择一新的中央候选巨图块$C_{x,y}$。新的中央巨图块是 C, TL, TR, BL, BR 候选组中在步骤 515 中算出有最低绝对差值加总值的方块。接着，在步骤 555，从梯度g_x与g_y计算新的步骤值Δ_x与Δ_y，例如$\Delta_x=\Delta_x\times g_x$。陡峭的梯度代表可接受的相符巨图块是目前中央候选很远，故增加(Δ_x,Δ_y)。相反地，浅的梯度代表可接受的相符巨图块是目前中央候选很近，故应减少(Δ_x,Δ_y)。熟悉此项技艺的人士应当了解到各种不同的系数可以从各梯度用来计算(Δ_x,Δ_y)以达成该结果。

接着，在步骤 560 测试迭代循环数。若该数目大于一最大值，则步骤 440 在步骤 565 完成，找不到可以接受的相符。此外，采用错误梯度以选择一组新的候选巨图块，其被期待为较接近于最终相符，该梯度下降步骤 440 回到步骤 510，在此产生一组新的。共轭梯度下降步骤 440 在以下两种情况下完成，当找到可接受的值（步骤 545），或最大迭代数目以达到仍无相符（步骤 565）。

图 6 说明使用共轭梯度下降步骤 440 的示范状态。初始候选巨图块$C_{x,y}$是方形（610C），而四个周围候选系圆圈（610T, 610L, 610R, 610B）。从这些初始候选计算梯度g_x与g_y(620X, 620Y)。在此示范状态中，梯度太浅了，而没有绝对差值加总值低于该临界值。因此延伸搜寻，使用四个新的中央候选巨图块，示为三角形（630TL, 630TR, 630BL, 630BR）。这些新的候选巨图块距离原本候选巨图块$C_{x,y}$周围角落Δ的距离。

在这些中央候选周围的巨图块，示为六角形（$640L_1$,$640T_1$,$640T_2$,$640R_2$,$640L_3$,$640B_3$,$640B_4$,$640R_4$），被选为候选。在此示范状态中，两个候选 640 具有低于临界值的绝对差值加总值与“陡峭”梯度（650XY,660XY）。另一候选根据各“陡峭”梯度选择：候选 670 根据梯度 650XY，而后选 680 是根据梯度 660XY。梯度下降搜寻继续使用这些新的候选 670、680，根据共轭梯度下降步骤 440。

d. 使用先前邻近帧间预测运动向量

图 7 是图 4 邻近搜寻算法（步骤 430）的流程图，由运动估测器 220 的一实施例所执行。如前所述，该搜寻的候选巨图块包含邻近于目前图像 205 中的目前巨图块 310（已被编码）的巨图块。亦包含为一候选的是在预先编码的参考图像 230 中的一对应巨图块。

计算候选巨图块坐标的步骤从步骤 710 开始，在此藉由利用目前巨图块 310 地址的绝对值（余数）与每行巨图块数计算一标记变量 TOPVALID。若此绝对值非 0，则 TOPVALID 为真，此外，TOPVALID 为假。在步骤 720，一标记变量 LEFTVALID 是利用目前巨图块 310 地址的除以整数与每行巨图块数计算。若此除数非 0，则 LEFTVALID 为真，此外，LEFTVALID 为假。这些 TOPVALID 与 LEFTVALID 变量表示目前巨图块 310 分别在上面与左边有一邻近巨图块，考虑巨图块的上缘与左边缘。

在步骤 730，结合使用 TOPVALID 与 LEFTVALID 变量以判定目前巨图块 310 邻近的 4 个候选巨图块的可得性，或存在性。特别是：左边有一巨图块 L 若（LEFTVALID）；上面有一巨图块 T 若（TOPVALID）；左上有一巨图块 TF 若（TOPVALID& LEFTVALID）；又上有一巨图块 TR 若（TOPVALID& RIGHTVALID）。接着，在步骤 740，为一先前候选巨图块 P 判定可得性，这是在空间上对应目前巨图块 310 的先前编码参考图像 230 中的一巨图块。这 5 个候选巨图块的相对位置可在图 8 中看到，其中，L 是 810、T 是 820、TL 是 830、TR 是 840、P 是 850。

回到图 7，步骤 730 与步骤 740 有多少候选巨图块可用来比较（从 1 到 5）。步骤 750 为每一可得候选巨图块计算绝对差值加总。若 5 个候选均可得，该组绝对差值加总值为：

$$\left\{0,L,T,P\left(\frac{L+T}{2}\right),med(L,T,TL),\left(\frac{L+med(T,TL,TR)}{2}\right),med(T,TL,TR)\right\}$$

若某些候选不可得，熟悉此项技艺的人士应当了解到该组候选相对较小。接着完成步骤 430，回复有最低绝对差值加总的候选巨图块。

如先前结合图 4 所讨论的，一旦找到相符巨图块（不论使用第图的邻近搜寻法或是图 5 的共轭梯度下降），接着搜寻区域更加限缩，采用局部彻底搜寻（图 4440）。在局部搜寻之后，利用局部彻底搜寻的结果计算一分数运动向量。分数运动向量的计算将在下面详述。

e. 利用二次表面模型的分数运动向量运算

熟悉此项技艺的人士应当对图示巨图块对搜寻窗间相符程度以产生“错误表面”感到熟悉。采用一开创性方法，运动估测器 220 以一二次表面建立错误表面的模型并分析地以次像素准确性判定该表面的最小值。运动估测器 220，首先判定一方所述的最小值，给定一最小行。运动估测器 220 接着沿着这条线决定正交方向的最小值。

二次曲线的一般方程式如方程式 1。

$$y = C_1 + C_2 t + C_3 t^2 \qquad \text{方程式 1}$$

对该曲线取微分，如第 2 方程式:

$$\frac{\delta y}{\delta t} = C_2 + 2C_3 t \Rightarrow t = \frac{-C_2}{2C_3} \qquad \text{方程式 2}$$

一旦系数C_1, C_2, C_3已知，则可求解以判定 t，最小的位置。运动估测器 220 解出方程式 3 以判定系数C_1, C_2, C_3。

$$\begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} = \frac{1}{4}\begin{pmatrix} 31 & -27 & 5 \\ -27 & 25 & -5 \\ 5 & -5 & 1 \end{pmatrix} \times \begin{pmatrix} \sum_{i=1}^{4} d_i \\ \sum_{i=1}^{4} d_i t_i \\ \sum d_i t_i^2 \end{pmatrix} \qquad \text{方程式 3}$$

运动估测器 220 使用由图形处理单元 120 所提供的 84 绝对差值加总指令已有效率的计算方程式 3。各d_i代表一绝对差值加总值，对 i 累加代表在 x 方向邻近巨图块的绝对差值加总值。如结合第 x 图的详细说明，该 8x4 绝对差值加总指令有效率的计算邻近的巨图块（x, y）、（x+1, y）、（x+2, y）、（x+3, y），的 4 个绝对差值加总值，即 i=0…3 且 i=j, t=j+1。如前所述，一旦系数已知，解方程式 2 得到 t，x 方向的最小值。

方程式 3 可以用来判定垂直方向的最小值 t。在此例中，运动估测器 220 使用 8x4 绝对差值加总指令已有效率的计算垂直地邻近的巨图块（x, y）、

(x+1, y)、(x+2, y)、(x+3, y)的4个绝对差值加总值。方程式3解出计算自这些绝对差值加总值的系数C1、C2、C3。如前所述，一旦系数已知，解方程式2得到t，y方向的最小值。运动估测器220所使用的二次错误表面方法较在先判断一像素边界上一较佳相符后再使用运算昂贵滤波器去寻找子像素边界上较佳相符的现有方法来的进步。

f. 在图形处理单元上使用绝对差值加总加速器以有效率的计算最小值

如前所述，运动估测器220以目前图像中一参考巨图块判定预测图像中那个巨图块有较佳的相符。运动估测器220使用由图形处理单元120所提供的绝对差值加总硬件加速，其为图形加速单元指令。绝对差值加总指令要输入一4x4参考方块与一8x4预测方块，并产生4个绝对差值加总值。参考方块与预测方块的大小可根据需要而改变。4x4参考方块与8x4预测方块仅为范例以说明本发明，而不应限制参考方块与预测方块的大小。图9A、B说明了对参考与预测方块进行绝对差值加总指令运作的方块图。如示于图9A，8x4预测方块由多个彼此重迭的水平邻近4x4方块所组成，如方块910、920、930、940。绝对差值加总单元取一个输入4x4参考方块950并计算该参考方块与910-940个方块的绝对差值加总值。即，该绝对差值加总指令计算4个值：一个值是方块910与方块950的差值的绝对值的总和；另一个值是方块920与方块950的差值的绝对值的总和；另一个值是方块930与方块950的差值的绝对值的总和；另一个值是方块940与方块950的差值的绝对值的总和。

参见图9B，图形处理单元120内的绝对差值加总加速单元使用4个绝对差值加总计算单元(960, 970, 980, 990)以实现绝对差值加总指令。最左边的4x4方块910被提供给绝对差值加总计算单元960。接着输入右边的4x4方块(920)给绝对差值加总计算单元970。接着输入右边的4x4方块(930)给绝对差值加总计算单元980。最后，提供最右边的4x4方块940给绝对差值加总计算单元990。图形处理单元120平行地使用独立的绝对差值加总计算单元，所以绝对差值加总指令每个周期产生4个绝对差值加总值。熟悉此项技艺的人士应当了解到用来计算两个相同大小像素方块的绝对差值加总运算的算法，以及用来执行此运算的硬件设计，故这些细节将不再详述。

4x4参考方块水平地且垂直地列在像素边缘。然而，不需要垂直地校正4x4预测方块910-940。在一实施例中，数据藉由旋转(逻辑电路995)该参考方块所校正。旋转参考方块而非分别旋转4个预测方块可节省逻辑门数。

旋转后的参考方块被提供给各独立绝对差值加总硬件加速单元。各单元产生12 位的值，而这些值结合成一个 48 位的输出。在一实施例中，这些值的数量级是根据预测方块的 U 纹理坐标（最低位位置中的最低坐标）。

下面的程序代码说明 8x8 方块，即两个邻近的 8x4 方块，的绝对差值加总值可以仅使用 4 个绝对差值加总指令计算。缓存器 T、T、T、T4 用来寄存这 4 个绝对差值加总值。变量 sadS 用来累加这些绝对差值加总值。8x4 参考方块的地址假设在 refReg。U 与 V 是 8x8 预测方块的纹理坐标。下面的程序代码产生整个 8x8 方块的全部的绝对差值加总值，存储在 sadS。

```
SAD   T1, refReg, U, V    ; left-top of 8x8 prediction block
SAD   T2, refReg, U+4, V     ; right-top of 8x8 prediction block
ADD   sadS, T1, T2
SAD   T3, refReg, U, V+4     ; left-bottom of 8x8 prediction
block
ADD   sadS, sadS, T3
SAD   T4, refReg, U+4, V+4   ; right-bottom of 8x8 prediction
block
ADD   sadS, sadS, T4
```

然而，通常可以避免计算与加总所有 4 个子方块的值，因为只要该总和达到目前最小值就可以停止该计算。下列的伪码说明如何在一循环内使用绝对差值加总指令，其在总和达到一最小值时停止。

```
I := 0;
SUM := 0;
MIN = currentMIN;
WHILE ( I < 4 || SUM < MIN)
    SUM := SUM + SAD(refReg, U+(I%2)*4, V+ (I>>1)*4);
IF (SUM < currMIN) currMIN = MIN;
Go to Next Search point;
```

图形处理单元 120 中的 84 绝对差值加总指令直接由运动估测器 220 的先进搜寻算法所使用，例如图 5 中所说明的执行局部彻底搜寻。此外，纹理快取 1060（图 10）是方块校正，而运动估测器 220 所使用的算法，如上所述，是像素校正。尽管可以将多路复用器单元加到图形处理单元 120 中以处理这

些校正误差，然而这么做会增加逻辑门数与电力消耗。取而代之，图形处理单元 120 使用这些多余的预算到 4 个绝对差值加总单元，而不是只用 1 个。在一些实施例中，8x4 绝对差值加总指令提供了有效率地运算最小值的优点，这牵涉到计算邻近方块的绝对差值加总值。在一些实施例中，8x4 绝对差值加总指令提供了彻底搜寻（方块 440）的另一优点，当步骤值为 1 时，其计算各对角的绝对差值加总值。

4. 图形处理器

已经讨论过运动估测器 220 的软件算法实现以及该算法在图形处理单元 120 中的 8x4 绝对差值加总指令的使用，接下来详细说明绝对差值加总指令与图形处理单元 120。

a. 图形处理单元流

图 10 是图形处理单元 120 的数据流程图，其中，指令流是由图 10 左边的箭头，而图像或图形流是由右边的箭头表示。图 10 省略了数个熟悉此项技艺者现有的元件，这些对解释图形处理单元 120 的回路内去方块效应特征非必要。

一指令流处理器 1010 从一系统总线（未示）接收一指令 1020，并解码该指令，产生指令数据 1030，例如顶点数据。图形处理单元 120 支持一现有图形处理指令，以及加速视频编码及/或解码的指令，例如前述的 8x4 绝对差值加总指令。

现有图形处理指令牵涉到如顶点着色（vertex shading）、几何着色（geometry shading）、像素着色（pixel shading）等难题。因此，指令数据 1030 施用于着色器执行单元（shader execution units）之池（pool）740。着色执行单元必要使用一纹理滤波单元（TFU，texture filter unit）750 以施加一纹理至一像素。纹理数据是快取自纹理快取 1060，其是在主存储器（未示）后面。

一些指令送给视频处理单元 1100，其运作将在后面说明。产生的数据接着由后包装器（post-packer 1070）处理，其压缩该数据。在后处理（post-processing）之后，由视频加速单元所产生的数据被提供给执行单元池（execution unit pool）1040。

视频编码/解码加速指令的执行，例如前述的绝对差值加总指令，在许多方面与前述的现有图形指令不同。首先，视频加速指令由视频处理单元 1100

执行，而非着色器执行单元。其次，视频加速指令不使用其纹理数据。

然而，视频加速指令所使用的图像数据与图形指令所使用的纹理数据均为 2 维阵列。图形处理单元 120 同样利用此优点，使用纹理滤波单元 1050 下载给视频处理单元 1100 的图像数据，因而使纹理快取 1060 快取一些由视频处理单元 1100 运作的图像数据。因此，示于图 10，视频处理单元 1100 位于纹理滤波单元 1050 与后包装器 1070 之间。

纹理滤波单元 1050 检验从指令 1020 撷取的指令数据 1030。指令数据 1030 更提供纹理滤波单元 1050 主存储器（未示）内想要的图像数据的坐标。在一实施例中，这些坐标标明为 U、V 对，熟悉此项技艺者应对此熟悉。当指令 1020 是一视频加速指令时，所撷取的指令数据 1030 更命令纹理滤波单元 1050 略过纹理滤波单元 1050 内的任何纹理滤波器（未示）。因此，纹理滤波单元 1050 受到视频加速指令的控制下载图像数据给视频处理单元 1100。

依此法，纹理滤波单元 1050 受操纵为视频加速指令去下载图像数据给视频加速单元 1100。视频处理单元 1100 从数据路径上的纹理滤波单元 1050 接收图像数据，与命令路径上的命令数据 1030，并根据命令数据 1030 对该图像数据执行一运作。由视频处理单元 1100 所输出图像数据被反馈给执行单元池 1040，在由后包装器 1070 处理之后。

b. 指令参数

现在说明视频处理单元 1100 在执行绝对差值加总视频加速指令的运作。如先前说明的，各图形处理单元指令解码且分析（parsed）为指令数据 1030，其可视为各指令的特定参数集。绝对差值加总指令的参数示于第 1 表。

第 1 表：图形处理单元的绝对差值加总指令

输入/输出	名称	大小	叙述
输入	FieldFlag	1-位	若 FieldFlag == 1 则 Field Picture，其余则 Frame Picture
输入	TopFieldFlag	1-位	若 TopFieldFlag == 1 则 Top-Field-Picture， 其它 Bottom-Field-Picture 若设定了 FieldFlag.

输入	PictureWidth	16-位	例如: 1920 用于 HDTV
输入	PictureHeigh t	16-位	例如: 1080 用于 30P HDTV
输入	BaseAddress	32-位无符号的	预测图片基本地址
输入	BlockAddress	U: 16-位有符号的 V: 16-位有符号的	预测图片纹理坐标 (关系于基本地址) 在 SRC1 Opcode SRC1[0:15] = U, SRC1[31:16] = V U,V 为 13.3 格式, 忽略分数部分
输入	RefBlock	128-位	参考图片数据 在 SRC2 Opcode
输出	Destination perand	4x16-位	128 位缓存器中最不重要的 32 位 在 DST Opcode

结合使用数个输入参数以判定由纹理滤波单元 1050 所撷取的 4x4 方块地址。BaseAddress 参数指出在纹理快取中该纹理数据的起点。将此区域内左上方块坐标给 BaseAddress 参数。PictureHeight 与 PictureWidth 输入参数用来判断该方块的范围，即左下方坐标。最后，视频图形可为逐行扫描(progessive)或隔行扫描(interlace)。若为隔行扫描，其由两个方向组成(上方与下方)。纹理滤波单元 750 使用 FieldFlag 与 TopFieldFlag 以适当处理隔行扫描图像。

c. 图像数据转换

为执行绝对差值加总指令，视频处理单元 1100 从纹理滤波单元 1050 撷取输入像素方块并对这些方块执行转换，转换为一适当格式以利绝对差值加总加速单元 960-990 处理。像素方块接着被提供至绝对差值加总加速单元 960-990，其回复绝对差值加总值。各绝对差值加总值接着被累积至目标缓存器。这些功能将在后面详述。

视频处理单元 1100 接收定义计算该绝对差值加总值的 8x4 方块的两个输入参数。参考方块的数据直接由 SRC2 运作码直接定义: 8x4x8 位方块视为 128 位的数据。相对地，SRC1 运作码定义预测方块的地址而非数据。视频处理单元 1100 提供这些地址给纹理滤波单元 1050，其从纹理快取 1060 撷取 128 位的预测方块数据。

尽管图像数据包含亮度（Y）与彩度（Cb,Cr）平面，运动估测通常仅使用 Y 成分。因此，当执行绝对差值加总指令时，视频处理单元 1100 所运作的像素方块仅含有 Y 成分。在一实施例中，视频处理单元 1100 产生一禁止信号，其指挥纹理滤波单元 1050 不要从纹理快取 1060 撷取 Cr/Cb 像素数据。

图 11 系纹理滤波单元 1050 与纹理快取 1060 的方块图。纹理滤波单元 1050 被设计为从纹理快取 1060 撷取纹理图像边界（texel boundry），并从纹理快取 1060 下载 4x4 纹理图像方块至滤波输入缓冲器 1110。当撷取数据代表视频处理单元 1100 时，纹理图像 1120 被视为各有 32 位的 4 个信道（ARGB），对于 128 位的纹理图像大小。当为绝对差值加总指令撷取数据时，纹理滤波单元 1050 下载 8x4x8 位方块。

为处理校正的问题，该 8x4 方块被下载至两个 4x4 像素输入缓冲器（1110A 与 1110B）。视频处理单元 1100 所示用的图像数据可能被字节校正。然而，纹理滤波单元 1050 被设计为从外取撷取纹理图像边界。因此，当为视频处理单元 1100 撷取的数据时，纹理滤波单元 1050 可能需要撷取达 4 个环绕在各个 4x4 半方块的一特定字节校正 8x4 方块周围的纹理图像校正 4x4 方块。

该程序可在图 11 中看到，其中，左半 4x4 方块（目标方块 1130）对准在纹理图像边界上，不论在垂直方向或在水平方向。换言之，目标方块 1130 延伸两个像素图像。该目标方块 1130 的 U、V 地址定义 4x4-8 位的最左上角，字节校正方块。在此例中，纹理滤波单元 1050 判断图像 1140、1150、1160、1170 应被撷取以得到目标方块 1130。在判断后，纹理滤波单元 1050 撷取方块 1140-1170 并接着结合从方块 1140-1170 所按位选择的行与列，故目标方块 1130 的最左边 4x4 位被写入滤波缓冲器 1110B。熟悉此项技艺的人士应当知道如何使用多路复用器、移位器（shifter）、屏蔽位（mask bits）达成该结果，不管从纹理快取 1060 所撷取的 4x4 目标校正。

在图 11 所示的实施例，当目标方块 1130 包含一垂直纹理像素边界，该数据不会垂直地重新排列。当此情形发生时，下载至滤波缓冲器 1110A 与 1110B 的数据在垂直方向的顺序与在快取中原本的顺序不同。在此实施例中，视频处理班员 1100 必须垂直地重新排列（旋转）128 位参考方块数据以符合预测方块的顺序。在另一实施例中，在写入其中一滤波缓冲器 1110 之前，纹理滤波单元 1050 垂直地重新排列快取纹理图像数据以符合原本的快取顺序。

任何程序说明或流程图中的方块应被理解为表示模块、区段或部分程序

代码，其包含用于实现特定逻辑电路功能或程序中的步骤的一个或多个可执行的指令。熟悉软件部门的技艺者应当了解到，其它的实现方法亦包含在所揭露的范围内。在其它的实现方法中，各功能可不依所示或揭露的顺序执行，包含实质上同步进行或逆向进行，依所涉及的功能而定。

在此揭露的系统与方法可以软件、硬件或其结合实现。在一些实施例中，该系统及/或方法是以存在存储器中的软件实现，且由位于一计算装置中的适当处理器所执行（包含而不限于一微处理器、微控制器、网络处理器、可重新装配处理器、可扩充处理器）。在其它实施例中，该系统及/或方法是以逻辑电路实现，包含而不限于一可程序逻辑装置（PLD，programmable logic device）、可程序逻辑门阵列（PGA，programmable gate array）、现场可编程逻辑门阵列（FPGA，field programmable gate array）或特定应用电路（ASIC）。在其它实施例中，这些逻辑叙述是在一图形处理器或图形处理单元（GPU）完成。

在此揭露的系统与方法可被嵌入任何计算机可读媒体而使用，或连结一指令执行系统、设备、装置。该指令执行系统包含任何以计算机为基础的系统、含有处理器的系统或其它可以从该指令执行系统撷取与执行这些指令的系统。所揭露的文字“计算机可读媒体(computer-readable medium)”可为任何可以容纳、存储、沟通、传递或传送该程序作为使用或与该指令执行系统连结的工具。该计算机可读媒体可为，例如（非限制）为基于电子的、有磁性的、光的、电磁的、红外线的或半导体技术的一系统或传递媒体。

使用电子技术的计算机可读媒体的特定范例（非限制）可包含：具有一条或多条电性（电子）连接的线；一随机存取存储器（RAM，random access memory）；一只读存储器（ROM，read-only memory）；一可擦可编程只读存储器（EPROM 或闪存）。使用磁技术的计算机可读媒体的特定范例（非限制）可包含：可携带计算机磁盘。使用光技术的计算机可读媒体的特定范例（非限制）可包含：一光纤与一可携带只读光盘（CD-ROM）。

虽然本发明在此以一个或更多个特定的范例作为实施例阐明及描述，不过不应将本发明局限于所示的细节，然而仍可在不背离本发明的精神下且在申请专利范围均等的领域与范围内实现许多不同的修改与结构上的改变。因此，最好将所附上的申请专利范围广泛地且以符合本发明领域的方法解释，在随后的申请专利范围前提出此声明。

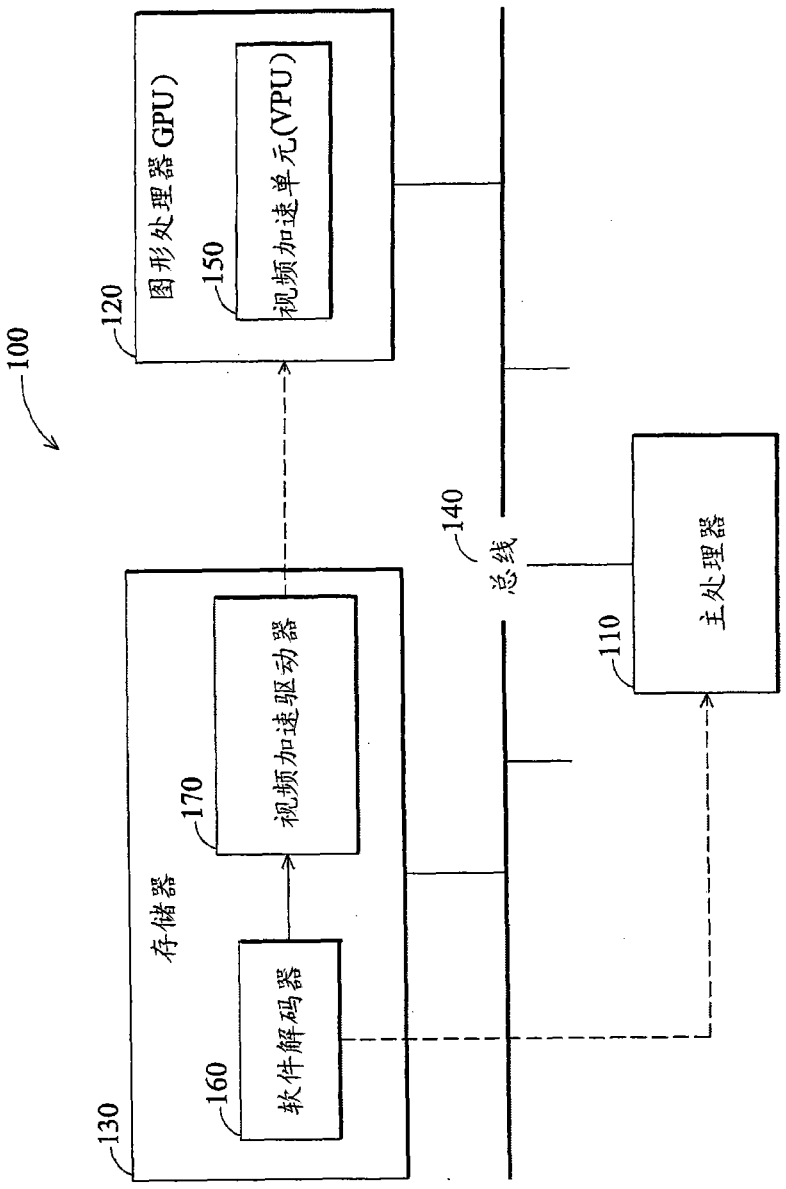

100
120
图形处理器 GPU)
150
视频加速单元(VPU)
140
总线
110
主处理器
130
存储器
170
视频加速驱动器
160
软件解码器

图 1

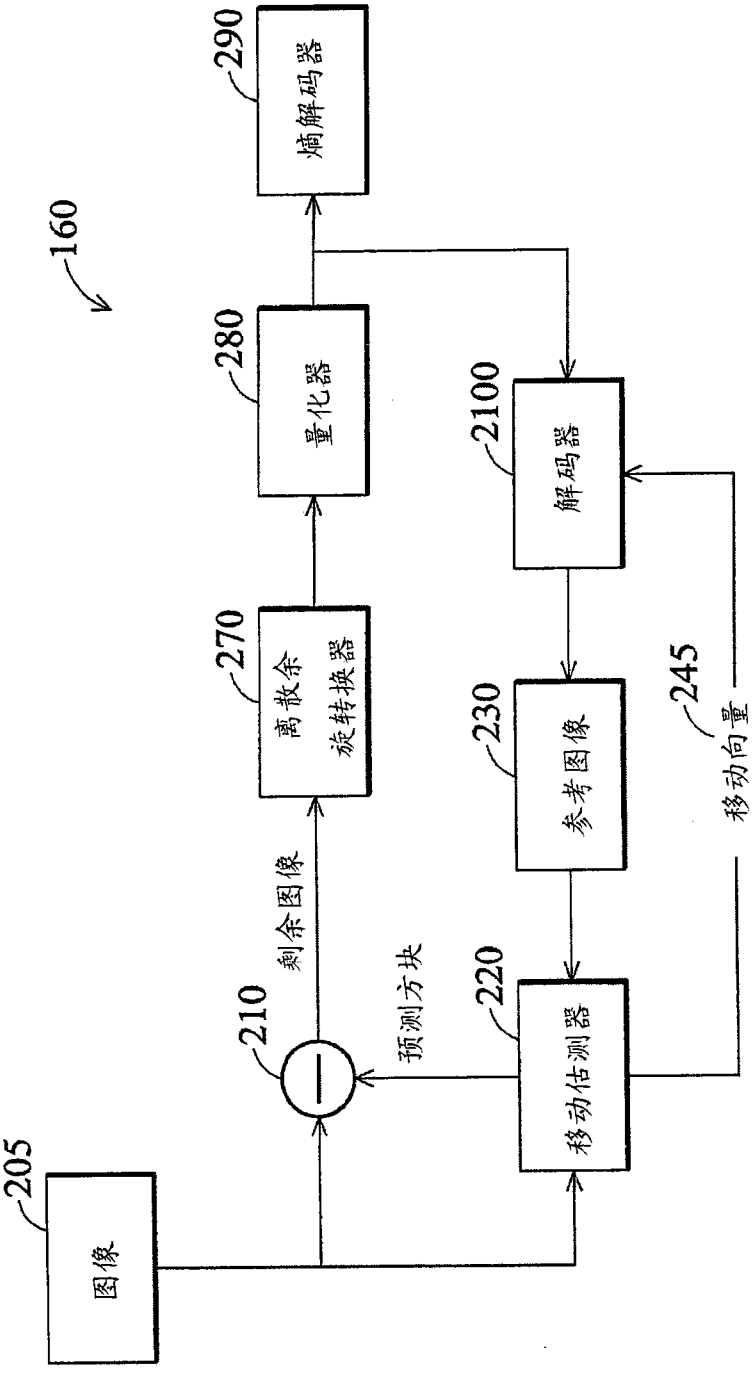

160
205
图像
210
剩余图像
预测方块
220
移动估测器
230
参考图像
245
移动向量
2100
解码器
270
离散余弦转换器
280
量化器
290
熵解码器

图 2

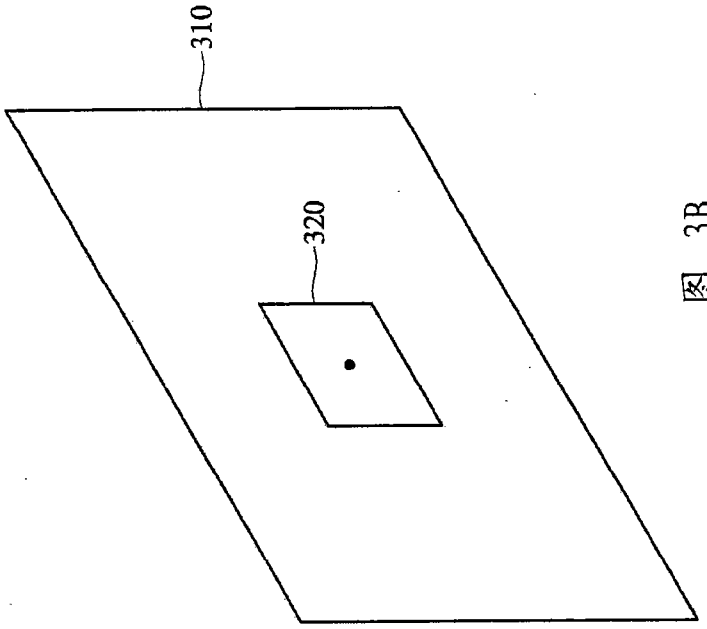


图 3B

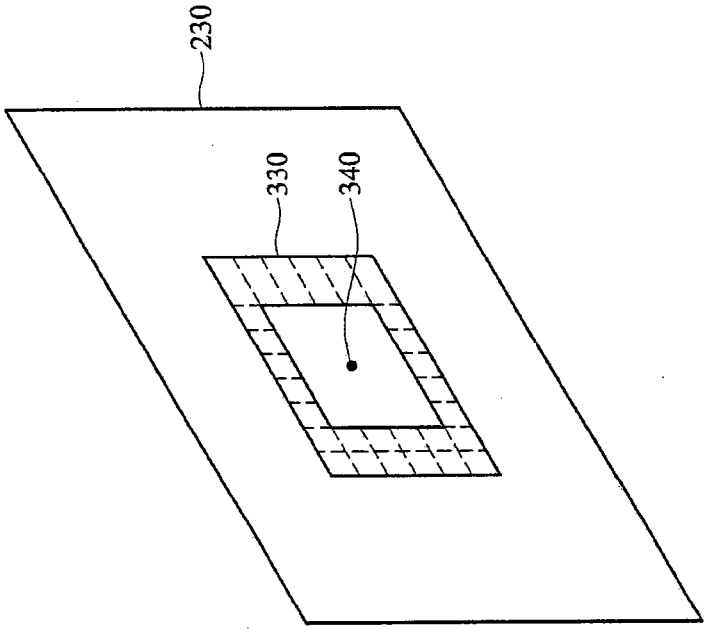


图 3A

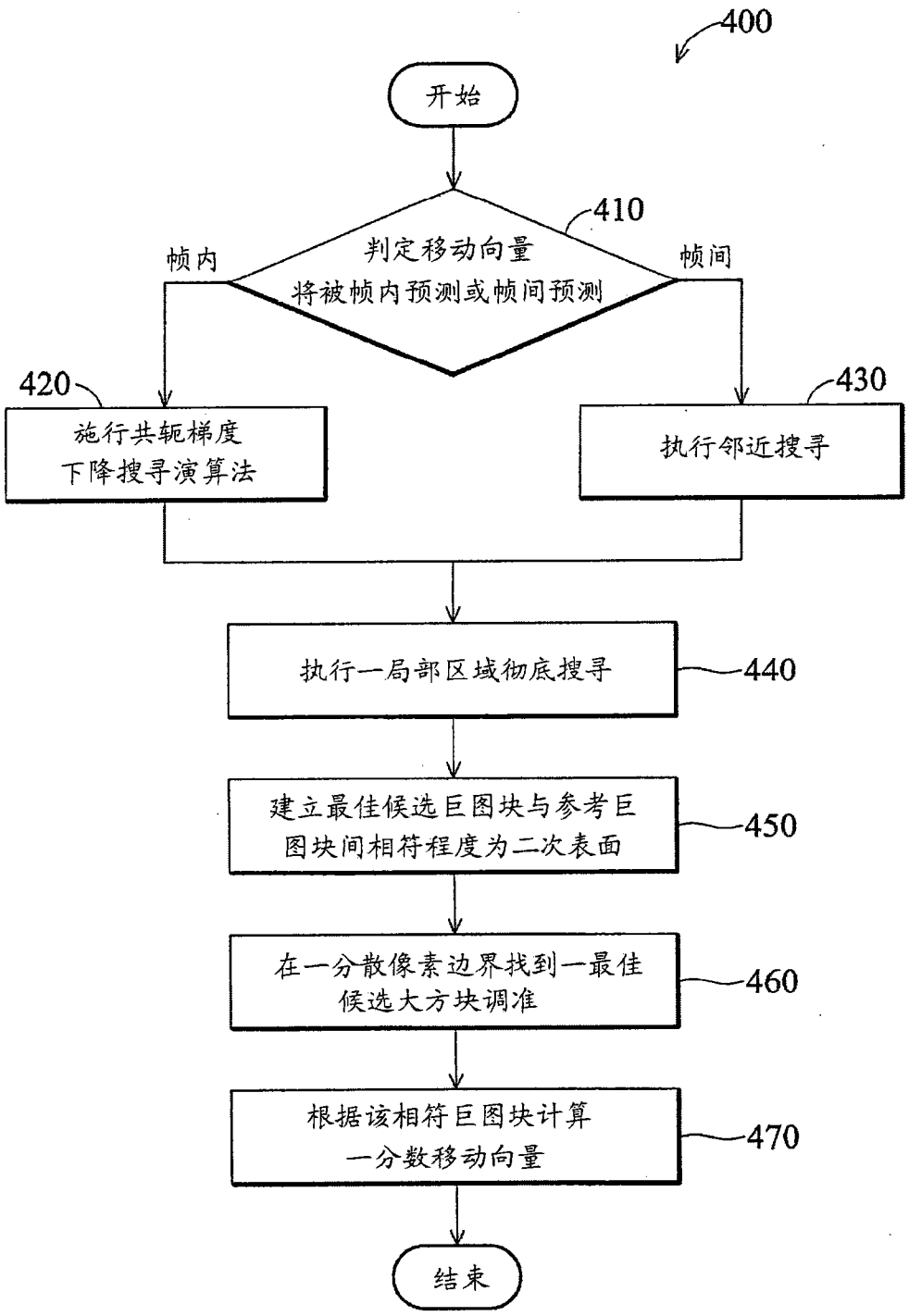

400
开始
410
判定移动向量
将被帧内预测或帧间预测
帧内
帧间
420
施行共轭梯度
下降搜寻演算法
430
执行邻近搜寻
执行一局部区域彻底搜寻
440
建立最佳候选巨图块与参考巨
图块间相符程度为二次表面
450
在一分散像素边界找到一最佳
候选大方块调准
460
根据该相符巨图块计算
一分数移动向量
470
结束

图 4

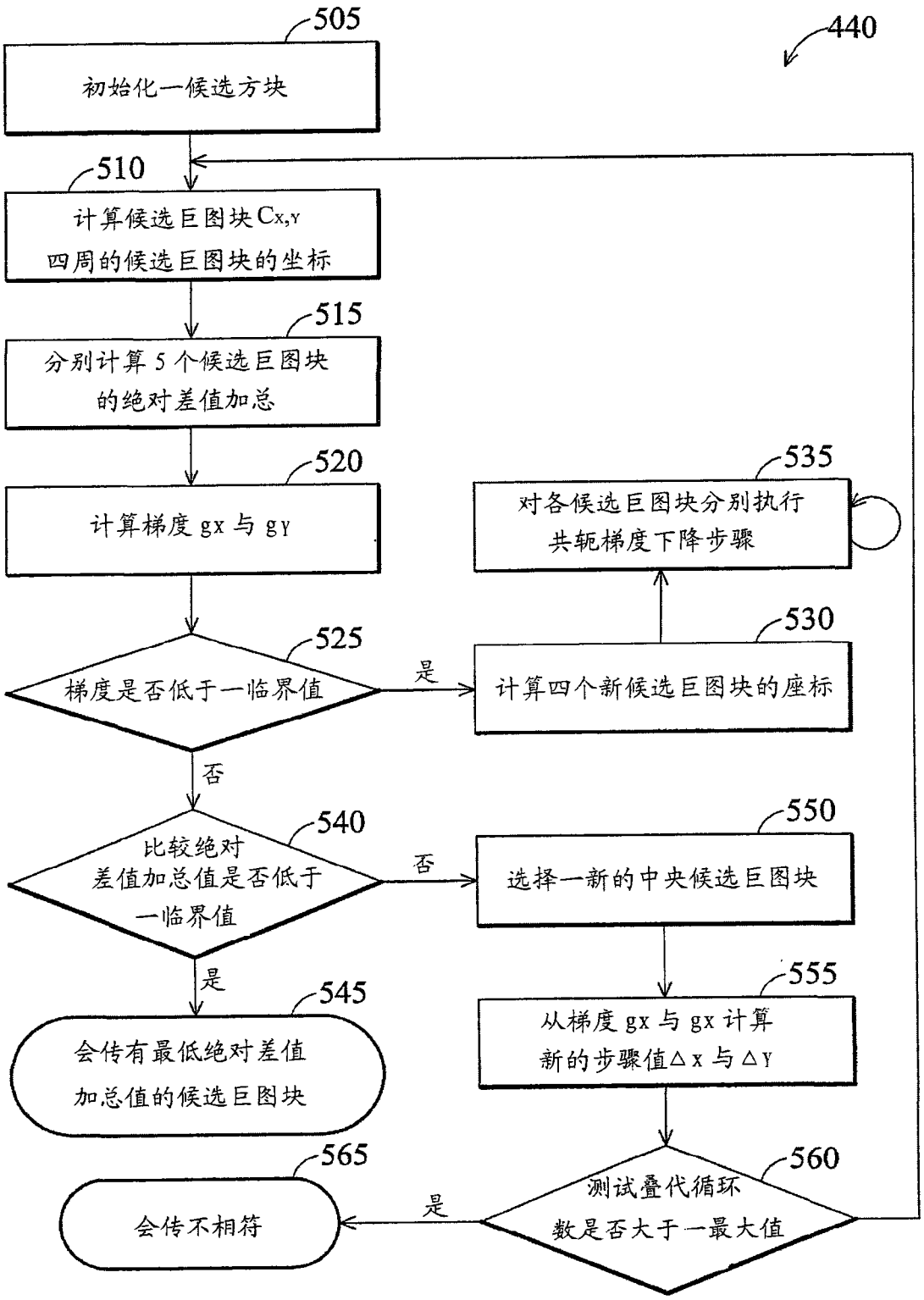

440
505
初始化一候选方块
510
计算候选巨图块Cx,y
四周的候选巨图块的坐标
515
分别计算5个候选巨图块
的绝对差值加总
520
计算梯度gx与gy
525
梯度是否低于一临界值
是
否
530
计算四个新候选巨图块的座标
535
对各候选巨图块分别执行
共轭梯度下降步骤
540
比较绝对
差值加总值是否低于
一临界值
否
是
545
会传有最低绝对差值
加总值的候选巨图块
550
选择一新的中央候选巨图块
555
从梯度gx与gx计算
新的步骤值Δx与Δy
560
测试叠代循环
数是否大于一最大值
是
565
会传不相符

图 5

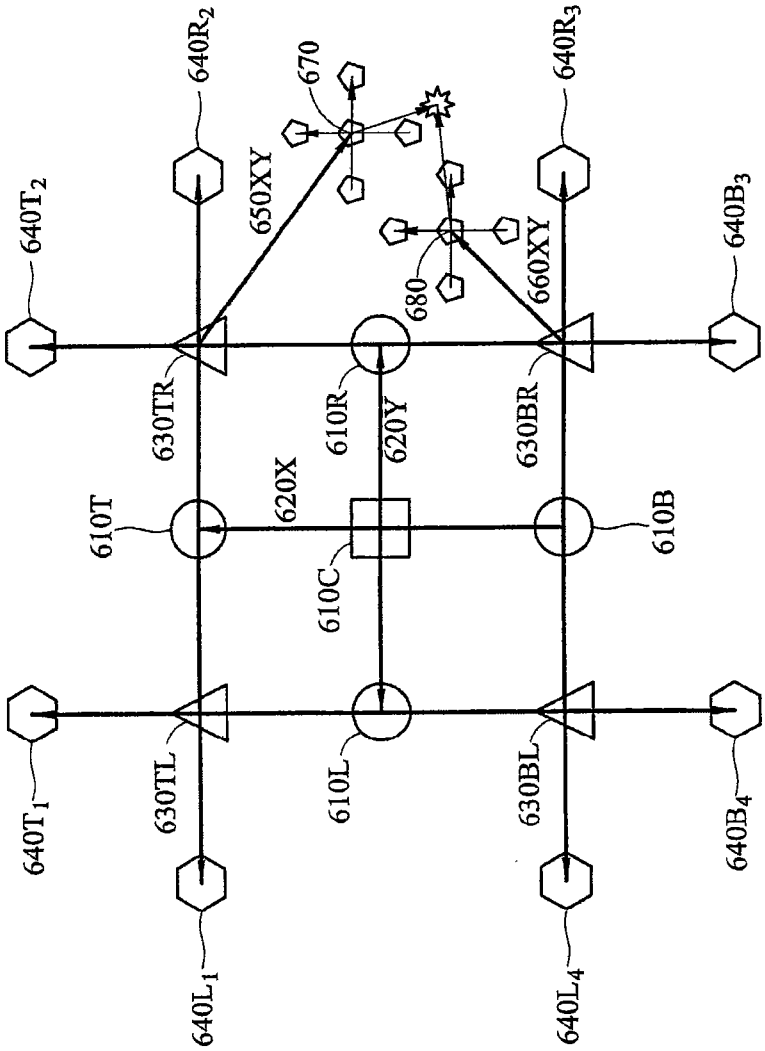

640R2
670
640R3
650XY
640T2
660XY
640B3
680
630TR
610R
620Y
630BR
620X
610T
610B
610C
630TL
610L
630BL
640T1
640B4
640L1
640L4

图 6

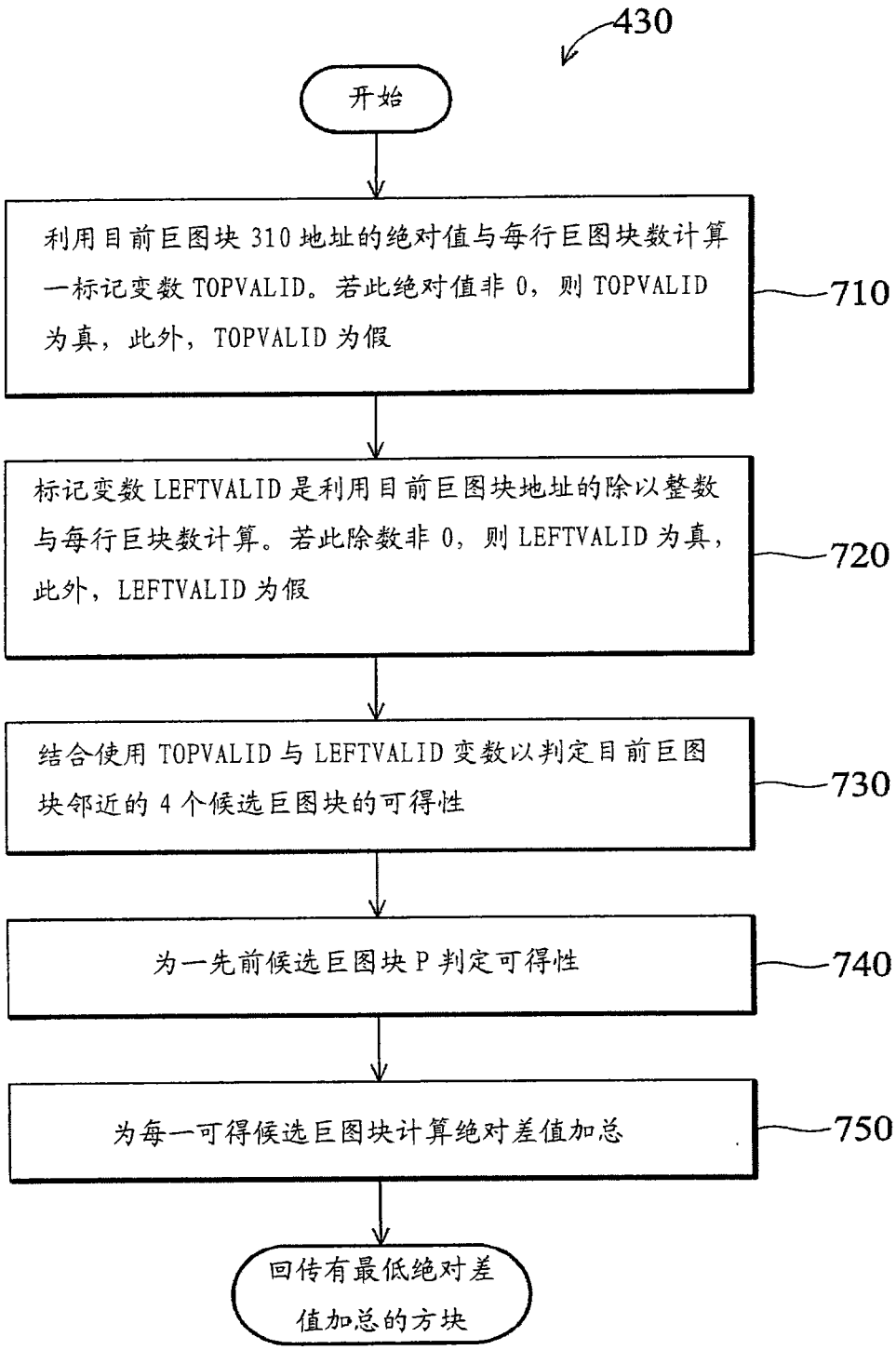

430
开始
利用目前巨图块 310 地址的绝对值与每行巨图块数计算一标记变数 TOPVALID。若此绝对值非 0，则 TOPVALID 为真，此外，TOPVALID 为假
710
标记变数 LEFTVALID 是利用目前巨图块地址的除以整数与每行巨块数计算。若此除数非 0，则 LEFTVALID 为真，此外，LEFTVALID 为假
720
结合使用 TOPVALID 与 LEFTVALID 变数以判定目前巨图块邻近的 4 个候选巨图块的可得性
730
为一先前候选巨图块 P 判定可得性
740
为每一可得候选巨图块计算绝对差值加总
750
回传有最低绝对差值加总的方块

图 7

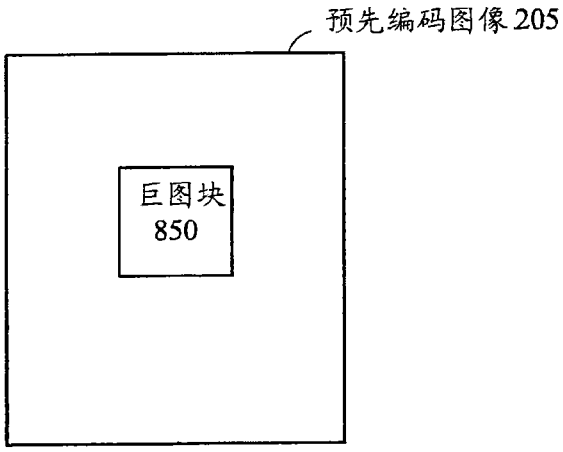


图 8A

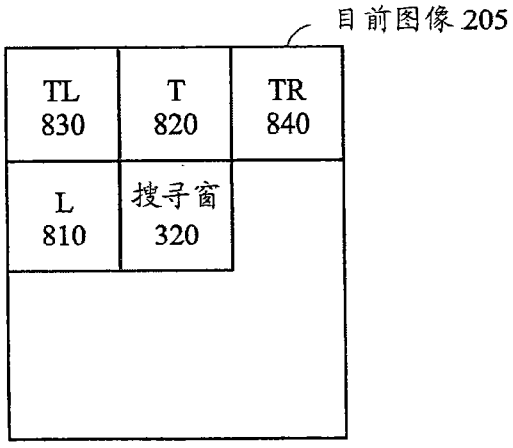


图 8B

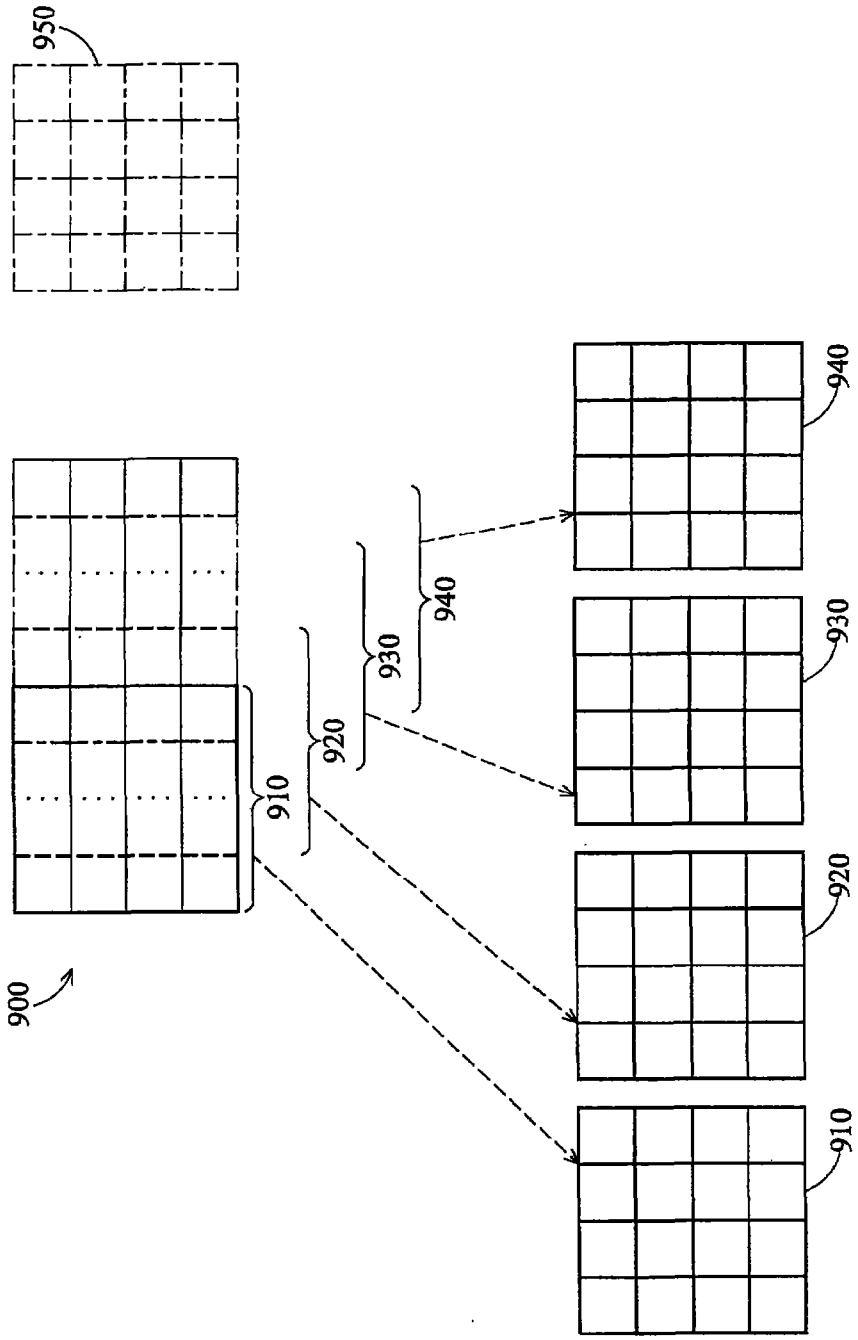


图 9A

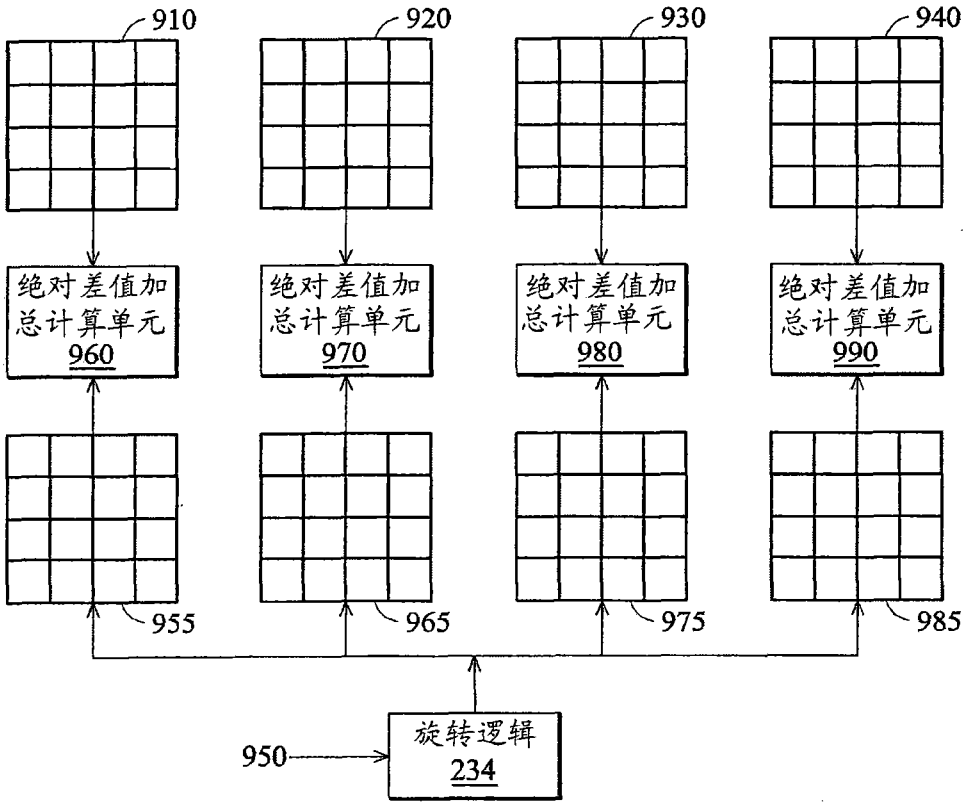

910
920
930
940
绝对差值加总计算单元 960
绝对差值加总计算单元 970
绝对差值加总计算单元 980
绝对差值加总计算单元 990
955
965
975
985
950
旋转逻辑 234

图 9B

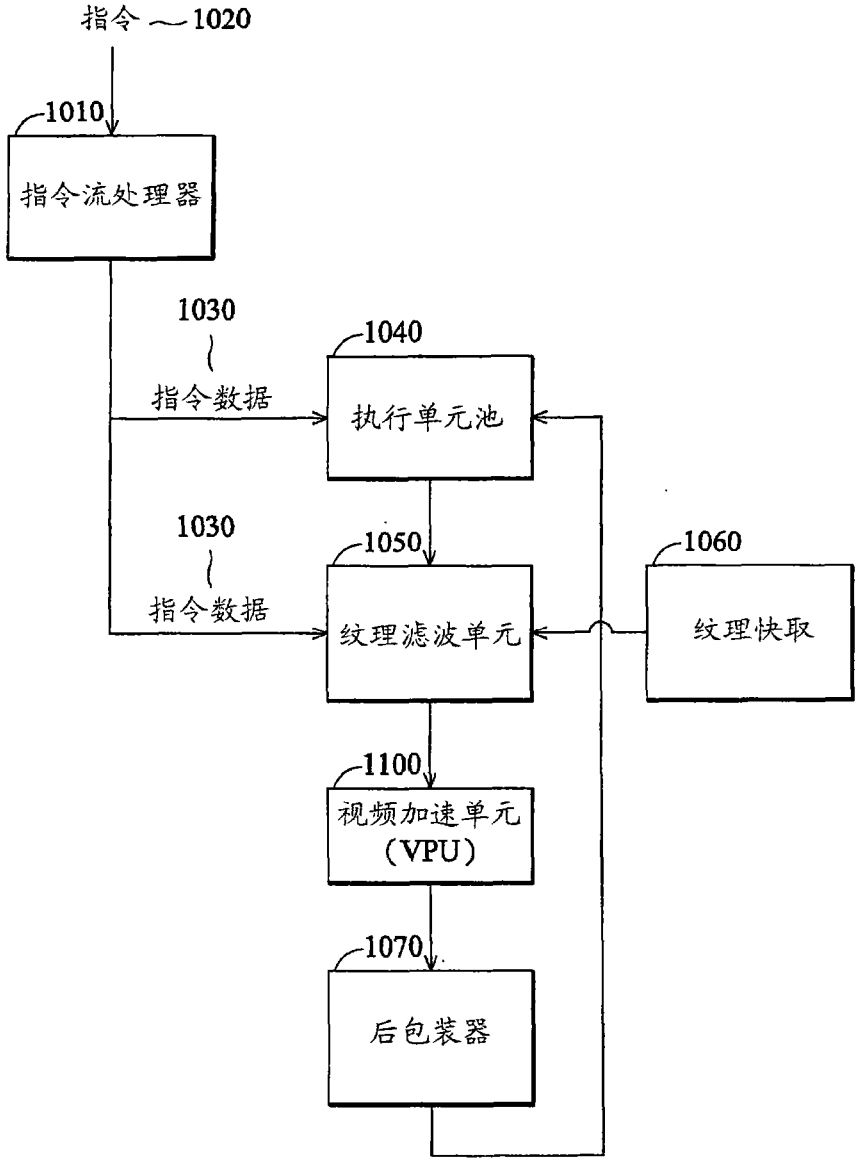

指令
1020
1010
指令流处理器
1030
指令数据
1040
执行单元池
1030
指令数据
1050
纹理滤波单元
1060
纹理快取
1100
视频加速单元
(VPU)
1070
后包装器

图 10

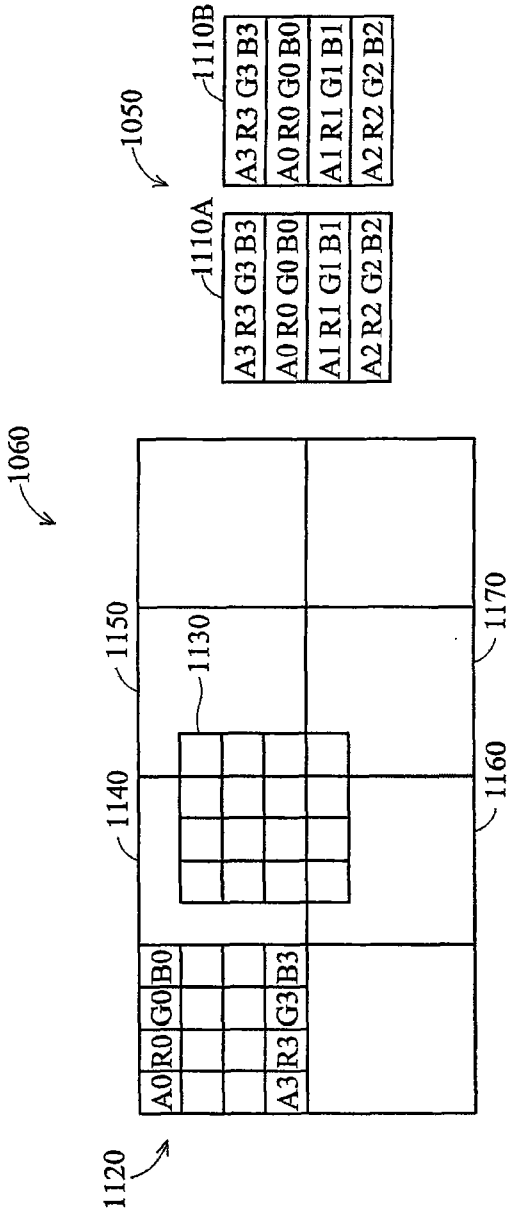

1050
1110B
A3 R3 G3 B3
A0 R0 G0 B0
A1 R1 G1 B1
A2 R2 G2 B2
1110A
A3 R3 G3 B3
A0 R0 G0 B0
A1 R1 G1 B1
A2 R2 G2 B2
1060
1150
1130
1170
1140
1160
A0 R0 G0 B0
A3 R3 G3 B3
1120

图 11