(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 107808394 A
(43)申请公布日 2018.03.16

(21)申请号 201711135318.0

(22)申请日 2017.11.16

(71)申请人 厦门美图之家科技有限公司
地址 361008 福建省厦门市湖里区火炬高新区创业园创业大厦11室

(72)发明人 李启东 李志阳 张伟 傅松林 洪炜冬

(74)专利代理机构 北京思睿峰知识产权代理有限公司 11396
代理人 谢建云 赵爱军

(51)Int.Cl.
G06T 7/41(2017.01)
G06N 3/04(2006.01)

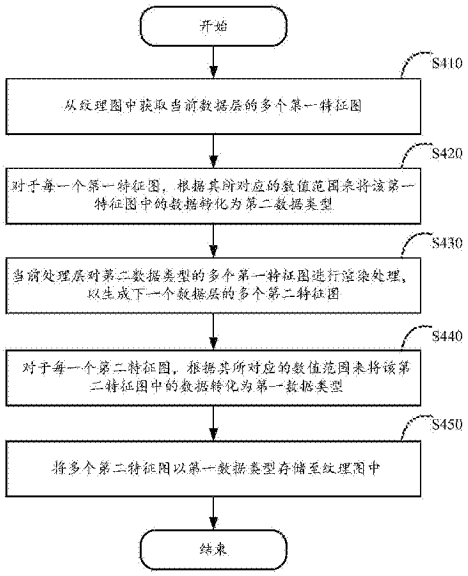
权利要求书3页 说明书15页 附图5页

(54)发明名称

一种基于卷积神经网络的图像处理方法及移动终端

(57)摘要

本发明公开了一种基于卷积神经网络的图像处理方法及移动终端，卷积神经网络包括多个数据层和处理层，该方法在移动终端的GPU中执行，GPU中存储有纹理图和网络参数，纹理图中以第一数据类型存储有当前数据层的多个第一特征图，网络参数包括每一个数据层的每一个特征图所对应的数值范围，该方法包括：从纹理图中获取当前数据层的多个第一特征图；对于每一个第一特征图，根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型；当前处理层对第二数据类型的多个第一特征图进行渲染处理，以生成下一个数据层的多个第二特征图；对于每一个第二特征图，根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型并存储至纹理图中。



CN 107808394 A

1.一种基于卷积神经网络的图像处理方法，所述卷积神经网络包括多个数据层和多个处理层，每个数据层包括多个特征图，所述方法在移动终端的图形处理器中执行，所述图形处理器包括图形存储器，所述图形存储器中存储有纹理图和网络参数，所述纹理图中以第一数据类型存储有当前数据层的多个第一特征图，所述网络参数包括每一个数据层的每一个特征图所对应的数值范围，所述方法包括：

从所述纹理图中获取当前数据层的多个第一特征图；

对于每一个第一特征图，根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型；

当前处理层对第二数据类型的多个第一特征图进行渲染处理，以生成下一个数据层的多个第二特征图；

对于每一个第二特征图，根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型；

将多个第二特征图以第一数据类型存储至所述纹理图中。

2.如权利要求1所述的方法，其中，每一个数据层的每一个特征图所对应的数值范围按照以下步骤确定：

将预定数量的测试图像输入所述卷积神经网络；

记录每一个测试图像在所述卷积神经网络的计算过程中所得到的每一个数据层的每一个特征图算例；

对于所述卷积神经网络的每一个数据层的每一个特征图，根据该特征图所对应的所有测试图像的特征图算例来确定该特征图的数值范围。

3.如权利要求2所述的方法，其中，所述预定数量的测试图像包括第一数量的真实图像和第二数量的随机图像，其中，所述第一数量大于等于所述第二数量，所述随机图像的RGB值为采用高斯分布函数随机生成。

4.如权利要求2或3所述的方法，其中，所述根据该特征图所对应的所有测试图像的特征图算例来确定该特征图的数值范围的步骤包括：

计算该特征图所对应的所有特征图算例的数据点均值的均值和标准差，所述特征图算例的数据点均值为特征图算例中各数据点的值的均值；

根据所述均值和标准差来确定该特征图的数值范围。

5.如权利要求4所述的方法，其中，所述计算该特征图所对应的所有特征图算例的数据点均值的均值和标准差的步骤包括：

对于第k个数据层的第j个特征图，分别计算该特征图所对应的每一个特征图算例的数据点均值$\mu_{kj,n}$和数据点平方均值$\mu^2_{kj,n}$，其中，k为数据层的编号，k大于等于1且小于等于所述卷积神经网络所包括的数据层的总数量K，j为特征图的编号，j大于等于1且小于等于第k个数据层所包括的特征图的总数量J(k)，n为特征图算例的编号，n大于等于1且小于等于测试图像的总数量N；

计算各特征图算例的数据点均值$\mu_{kj,n}$的均值$\mu_{kj}=\frac{1}{N}\sum_{n=1}^{N}\mu_{kj,n}$，和标准差

$$\sigma_{kj}=\sqrt{(\frac{1}{N}\sum_{n=1}^{N}\mu_{kj,n}^{2})-\mu_{kj}^{2}}\ ;$$

所述根据所述均值和标准差来确定该特征图的数值范围的步骤包括：

将该特征图的数值范围设置为(min_{kj},max_{kj})，其中，$min_{kj}=\mu_{kj}-3\sigma_{kj}$，$max_{kj}=\mu_{kj}+3\sigma_{kj}$。

6.如权利要求1-5中任一项所述的方法，其中，所述第一数据类型为八位无符号整型，所述第二数据类型为浮点型；

所述根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型的步骤包括：

将第一特征图中的数据由0～255范围内的整数归一化为0.0～1.0范围内的浮点数；

将所述0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数；

所述根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型的步骤包括：

根据其所对应的数值范围来将该第二特征图中的数据转化为0.0～255.0范围内的浮点数；

将所述0.0～255.0范围内的浮点数转化为小于等于其本身的最大整数。

7.如权利要求6所述的方法，其中，将当前数据层k的第j_1个第一特征图的数值范围记为$(\min_{kj_1},\ \max_{kj_1})$，将下一个数据层(k+1)的第$j_2$个第二特征图的数值范围记为$(\min_{(k+1)j_2},\ \max_{(k+1)j_2})$；

按照以下公式将所述0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数：

$$f_{kj_1}=sf_{kj_1}\times(\max_{kj_1}-\min_{kj_1})+\min_{kj_1}$$

其中，f_{kj1}为$(\min_{kj_1},\ \max_{kj_1})$范围内的浮点数，$sf_{kj1}$为0.0～1.0范围内的浮点数；

按照以下公式将第二特征图中的数据转化为0.0～255.0范围内的浮点数：

$$u_{(k+1)j_2}=\begin{cases}255.0, & f_{(k+1)j_2}>\max_{(k+1)j_2}\\ \dfrac{f_{(k+1)j_2}-\min_{(k+1)j_2}}{\max_{(k+1)j_2}-\min_{(k+1)j_2}}\times 255.0, & \min_{(k+1)j_2}\le f_{(k+1)j_2}\le\max_{(k+1)j_2}\\ 0.0, & f_{(k+1)j_2}<\min_{(k+1)j_2}\end{cases}$$

其中，$u_{(k+1)j_2}$为0.0～255.0范围内的浮点数，$f_{(k+1)j_2}$为下一个数据层(k+1)的第j_2个第二特征图中的任意一个数据点的值。

8.如权利要求1-7中任一项所述的方法，其中，所述纹理图包括多个纹理区块，每个纹理区块包括RGBA四个通道，每个通道适于存储一个第二特征图；

所述将多个第二特征图以第一数据类型存储至所述纹理图中的步骤包括：将多个第二特征图以第一数据类型按顺序存储至各纹理区块的各通道中。

9.一种移动终端，包括：

至少一个图形处理器；和

存储有程序指令的存储器，其中，所述程序指令被配置为适于由所述至少一个图形处理器执行，所述程序指令包括用于执行如权利要求1-8中任一项所述的基于卷积神经网络的图像处理方法的指令。

10.一种存储有程序指令的可读存储介质，当所述程序指令被移动终端读取并执行时，使得所述移动终端执行如权利要求1-8中任一项所述的基于卷积神经网络的图像处理方法。

一种基于卷积神经网络的图像处理方法及移动终端

技术领域

[0001] 本发明涉及图像处理技术领域，尤其涉及一种基于卷积神经网络的图像处理方法及移动终端。

背景技术

[0002] 卷积神经网络(CNN，Convolutional Neural Network)在图像处理领域扮演着重要的角色，例如，图像的分类、分割、风格转换、画质改善等，均可以采用CNN来实现，以取得比传统处理方法更好的效果。目前，基于CNN的图像处理在PC端得到了广泛应用，然而，其在移动终端上的应用仍存在瓶颈，尤其是当CNN达到数百层时，需要大量的浮点数乘法运算及大量的内存申请，导致移动终端的计算效率跟不上CNN的发展速度。

[0003] 移动端GPU的发展以及跨平台的OpenGL ES(Open Graphics Library for Embedded Systems)图形程序接口为移动端的CNN计算带来了极大的便利。虽然基于OpenGL ES的GPU并行计算提高了CNN的处理效率，在脚本渲染时能够快速进行浮点数的运算，但其在数据存储方面仍存在瓶颈。一方面，OpenGL ES的纹理支持的数据类型为无符号8位整型(uint8)，16位或32位浮点(float16，float32)等，但移动终端的存储空间参差不齐，存储空间较小的移动终端难以支撑CNN计算，OpenGL ES的渲染脚本未必适用于所有移动终端。另一方面，OpenGL ES支持的纹理的大小也有限制，相对低端的GPU芯片，纹理的宽和高均需限制在2048个像素的范围内。此外，对于图像风格转换、视频风格转换等复杂的图像处理，CNN往往多达几十层甚至数百层，在采用纹理存储CNN数据时，势必会导致大量纹理的创建、绑定和解绑，降低了计算效率。

[0004] 因此，需要提供一种兼容性更好、且计算效率更高的CNN图像处理方法。

发明内容

[0005] 为此，本发明提供一种基于卷积神经网络的图像处理方法及移动终端，以解决或至少缓解上面存在的问题。

[0006] 根据本发明的一个方面，提供一种基于卷积神经网络的图像处理方法，所述卷积神经网络包括多个数据层和多个处理层，每个数据层包括多个特征图，所述方法在移动终端的图形处理器中执行，所述图形处理器包括图形存储器，所述图形存储器中存储有纹理图和网络参数，所述纹理图中以第一数据类型存储有当前数据层的多个第一特征图，所述网络参数包括每一个数据层的每一个特征图所对应的数值范围，所述方法包括：从所述纹理图中获取当前数据层的多个第一特征图；对于每一个第一特征图，根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型；当前处理层对第二数据类型的多个第一特征图进行渲染处理，以生成下一个数据层的多个第二特征图；对于每一个第二特征图，根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型；将多个第二特征图以第一数据类型存储至所述纹理图中。

[0007] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，每一个数据层的

每一个特征图所对应的数值范围按照以下步骤确定：将预定数量的测试图像输入所述卷积神经网络；记录每一个测试图像在所述卷积神经网络的计算过程中所得到的每一个数据层的每一个特征图算例；对于所述卷积神经网络的每一个数据层的每一个特征图，根据该特征图所对应的所有测试图像的特征图算例来确定该特征图的数值范围。

[0008] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述预定数量的测试图像包括第一数量的真实图像和第二数量的随机图像，其中，所述第一数量大于等于所述第二数量，所述随机图像的RGB值为采用高斯分布函数随机生成。

[0009] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述根据该特征图所对应的所有测试图像的特征图算例来确定该特征图的数值范围的步骤包括：计算该特征图所对应的所有特征图算例的数据点均值的均值和标准差，所述特征图算例的数据点均值为特征图算例中各数据点的值的均值；根据所述均值和标准差来确定该特征图的数值范围。

[0010] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述计算该特征图所对应的所有特征图算例的数据点均值的均值和标准差的步骤包括：对于第k个数据层的第j个特征图，分别计算该特征图所对应的每一个特征图算例的数据点均值μ_{kjn}和数据点平方均值$\mu_{kj,n}^2$，其中，k为数据层的编号，k大于等于1且小于等于所述卷积神经网络所包括的数据层的总数量K，j为特征图的编号，j大于等于1且小于等于第k个数据层所包括的特征图的总数量J(k)，n为特征图算例的编号，n大于等于1且小于等于测试图像的总数量N；计算各特征图算例的数据点均值$\mu_{kj,n}$的均值$\mu_{kj}=\frac{1}{N}\sum_{n=1}^{N}\mu_{kjn}$，和标准差$\sigma_{kj}=\sqrt{(\frac{1}{N}\sum_{n=1}^{N}\mu_{kj,n}^2)-\mu_{kj}^2}$；所述根据所述均值和标准差来确定该特征图的数值范围的步骤包括：将该特征图的数值范围设置为(min_{kj},max_{kj})，其中，$min_{kj}=\mu_{kj}-3\sigma_k$，$max_{kj}=\mu_{kj}+3\sigma_{kj}$。

[0011] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述第一数据类型为八位无符号整型，所述第二数据类型为浮点型；所述根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型的步骤包括：将第一特征图中的数据由0～255范围内的整数归一化为0.0～1.0范围内的浮点数；将所述0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数；所述根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型的步骤包括：根据其所对应的数值范围来将该第二特征图中的数据转化为0.0～255.0范围内的浮点数；将所述0.0～255.0范围内的浮点数转化为小于等于其本身的最大整数。

[0012] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，将当前数据层k的第j_1个第一特征图的数值范围记为$(\min_{kj_1},\ \max_{kj_1})$，将下一个数据层(k+1)的第$j_2$个第二特征图的数值范围记为$(\min_{(k+1)j_2},\ \max_{(k+1)j_2})$；按照以下公式将所述0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数：

[0013] $$f_{kj_1}=sf_{kj_1}\times(\max_{kj_1}-\min_{kj_1})+\min_{kj_1}$$

[0014] 其中，$fkj1$为$(\min_{kj_1}, \max_{kj_1})$范围内的浮点数，$sfkj1$为0.0～1.0范围内的浮点数；按照以下公式将第二特征图中的数据转化为0.0～255.0范围内的浮点数：

[0015] $$u_{(k+1)j_2}=\begin{cases}255.0, & f_{(k+1)j_2}>\max_{(k+1)j_2}\\ \dfrac{f_{(k+1)j_2}-\min_{(k+1)j_2}}{\max_{(k+1)j_2}-\min_{(k+1)j_2}}\times 255.0, & \min_{(k+1)j_2}\le f_{(k+1)j_2}\le \max_{(k+1)j_2}\\ 0.0, & f_{(k+1)j_2}<\min_{(k+1)j_2}\end{cases}$$

[0016] 其中，$u_{(k+1)j_2}$为0.0～255.0范围内的浮点数，$f_{(k+1)j_2}$为下一个数据层(k+1)的第j_2个第二特征图中的任意一个数据点的值。

[0017] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述纹理图包括多个纹理区块，每个纹理区块包括RGBA四个通道，每个通道适于存储一个第二特征图；所述将多个第二特征图以第一数据类型存储至所述纹理图中的步骤包括：将多个第二特征图以第一数据类型按顺序存储至各纹理区块的各通道中。

[0018] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述网络参数还包括第二特征图的数量和尺寸，以及下一个数据层所对应的纹理图所包括的纹理区块的数量和纹理图的尺寸，其中，所述第二特征图的尺寸包括第二特征图的横向数据点的数量和纵向数据点的数量；所述纹理区块的数量为ceil(c/4)，其中，c为第二特征图的数量，ceil(c/4)表示大于等于(c/4)的最小整数；所述纹理图的尺寸按照以下步骤确定：将纹理区块的数量因数分解为w*h，以使得(w*第二特征图的横向数据点的数量)与(h*第二特征图的纵向数据点的数量)的差值的绝对值最小；所述纹理图的横向数据点的数量为(w*第二特征图的横向数据点的数量)，所述纹理图的纵向数据点的数量为(h*第二特征图的纵向数据点的数量)。

[0019] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述将多个第二特征图以第一数据类型按顺序存储至各纹理区块的各通道中的步骤包括：将第i个第二特征图以第一数据类型存储至第ceil(i/4)个纹理区块的第(mod(i-1,4)+1)个通道中，其中，mod(i-1,4)表示(i-1)除以4所得的余数。

[0020] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述移动终端还包括存储器，所述存储器中存储有待处理图像；所述方法还包括：将存储器中的所述待处理图像缓存至所述图形存储器中，并将所述待处理图像作为第一个数据层输入到所述卷积神经网络；所述卷积神经网络的第一个处理层对所述待处理图像进行渲染处理，以生成第二个数据层的多个特征图；对于第二个数据层的每一个特征图，根据其所对应的数值范围来将该特征图中的数据转化为第一数据类型；将第二个数据层的多个特征图以第一数据类型存储至所述纹理图中。

[0021] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，还包括：从所述纹理图中获取倒数第二个数据层的多个特征图；对于倒数第二个数据层的每一个特征图，根据其所对应的数值范围来将该特征图中的数据转化为第二数据类型；所述卷积神经网络的最后一个处理层对倒数第二个数据层的多个特征图进行渲染处理，以生成结果图像；将所述结果图像输出至所述存储器中进行存储。

[0022] 可选地，在根据本发明的基于卷积神经网络的图像处理方法中，所述渲染处理的步骤由OpenGL ES图形程序接口来完成，所述图形存储器中还存储有适于执行所述渲染处理的OpenGL ES脚本。

[0023] 根据本发明的另一个方面，提供一种移动终端，包括：至少一个图形处理器；和存储有程序指令的存储器，其中，所述程序指令被配置为适于由所述至少一个图形处理器执行，所述程序指令包括用于执行如上所述的基于卷积神经网络的图像处理方法的指令。

[0024] 根据本发明的又一个方面，提供一种存储有程序指令的可读存储介质，当所述程序指令被移动终端读取并执行时，使得所述移动终端执行如上所述的基于卷积神经网络的图像处理方法。

[0025] 根据本发明的技术方案，GPU的图形存储器中存储有纹理图，纹理图中以第一数据类型(uint8)存储数据，对于CNN的各处理层，先将纹理图中的数据转化为第二数据类型(float16或float32)，再对第二数据类型的数据进行渲染处理。第一数据类型(uint8)可以满足大部分移动终端的存储空间限制，使得本方案可以适用于绝大多数的移动终端，兼容性较好。

[0026] 本发明采用一张纹理图来存储当前数据层的多个第一特征图，纹理图中包括多个纹理区块，每个纹理区块包括RGBA四个通道，每个通道可以存储一张第一特征图。这样，所有的特征图数据均存储于一张纹理图中，在CNN的各处理层进行渲染处理时，从纹理图的各纹理区块的各通道中读取数据，进行分块渲染，无需创建多个纹理，避免了纹理渲染过程中不断地绑定、解绑纹理的操作，从而节省了计算时间，提高了计算效率。

[0027] 由于本发明的纹理图中存储了当前参与计算的所有特征图，且图形存储器中存储有CNN的各项网络参数以及OpenGL ES脚本，在GPU执行CNN计算时，可以直接从图形存储器中快速读写数据并进行渲染。除了待处理图像的输入以及结果图像的输出，各处理层的计算完全由GPU完成，而无需与CPU或内存进行数据交换，充分利用了GPU并行浮点数计算的优势，大大提高了计算效率，使得移动端的复杂图像处理(例如图像风格转换、视频风格转换等)成为可能。

[0028] 上述说明仅是本发明技术方案的概述，为了能够更清楚了解本发明的技术手段，而可依照说明书的内容予以实施，并且为了让本发明的上述和其它目的、特征和优点能够更明显易懂，以下特举本发明的具体实施方式。

附图说明

[0029] 为了实现上述以及相关目的，本文结合下面的描述和附图来描述某些说明性方面，这些方面指示了可以实践本文所公开的原理的各种方式，并且所有方面及其等效方面旨在落入所要求保护的主题的范围内。通过结合附图阅读下面的详细描述，本公开的上述以及其它目的、特征和优势将变得更加明显。遍及本公开，相同的附图标记通常指代相同的部件或元素。

[0030] 图1示出了根据本发明一个实施例的移动终端100的示意图；

[0031] 图2示出了根据本发明一个实施例的卷积神经网络的结构图；

[0032] 图3示出了根据本发明一个实施例的中央处理器103、图形处理器104、以及存储器150的示意图；

[0033] 图4示出了根据本发明一个实施例的基于卷积神经网络的图像处理方法400的流程图;以及

[0034] 图5示出了根据本发明一个实施例的纹理图的示意图。

具体实施方式

[0035] 下面将参照附图更详细地描述本公开的示例性实施例。虽然附图中显示了本公开的示例性实施例,然而应当理解,可以以各种形式实现本公开而不应被这里阐述的实施例所限制。相反,提供这些实施例是为了能够更透彻地理解本公开,并且能够将本公开的范围完整的传达给本领域的技术人员。

[0036] 图1示出了根据本发明一个实施例的移动终端100的示意图。移动终端100例如可以是手机、平板电脑、游戏机、多媒体设备、智能可穿戴设备等,但不限于此。如图1所示,移动终端100可以包括存储器接口102、图形处理器(GPU,Graphics Processing Unit)103、中央处理器(CPU,Central Processing Unit)104以及外围接口106。

[0037] 存储器接口102、图形处理器103、中央处理器104以及外围接口106既可以是分立元件,也可以集成在一个或多个集成电路中。在移动终端100中,各种元件可以通过一条或多条通信总线或信号线来耦合。传感器、设备和子系统可以耦合到外围接口106,以便帮助实现多种功能。

[0038] 例如,运动传感器110、光线传感器112和距离传感器114可以耦合到外围接口106,以方便定向、照明和测距等功能。其他传感器116同样可以与外围接口106相连,例如定位系统(例如GPS接收机)、温度传感器、生物测定传感器或其他感测设备,由此可以帮助实施相关的功能。

[0039] 相机子系统120和光学传感器122可以用于方便诸如记录照片和视频剪辑的相机功能的实现,其中所述相机子系统和光学传感器例如可以是电荷耦合器件(CCD)或互补金属氧化物半导体(CMOS)光学传感器。可以通过一个或多个无线通信子系统124来帮助实现通信功能,其中无线通信子系统可以包括射频接收机和发射机和/或光(例如红外)接收机和发射机。无线通信子系统124的特定设计和实施方式可以取决于移动终端100所支持的一个或多个通信网络。例如,移动终端100可以包括被设计成支持LTE、3G、GSM网络、GPRS网络、EDGE网络、Wi-Fi或WiMax网络以及Bluebooth™网络的通信子系统124。

[0040] 音频子系统126可以与扬声器128以及麦克风130相耦合,以便帮助实施启用语音的功能,例如语音识别、语音复制、数字记录和电话功能。I/O子系统140可以包括触摸屏控制器142和/或一个或多个其他输入控制器144。触摸屏控制器142可以耦合到触摸屏146。举例来说,该触摸屏146和触摸屏控制器142可以使用多种触摸感测技术中的任何一种来检测与之进行的接触和移动或是暂停,其中感测技术包括但不局限于电容性、电阻性、红外和表面声波技术。一个或多个其他输入控制器144可以耦合到其他输入/控制设备148,例如一个或多个按钮、摇杆开关、拇指旋轮、红外端口、USB端口、和/或指示笔之类的指点设备。所述一个或多个按钮(未显示)可以包括用于控制扬声器128和/或麦克风130音量的向上/向下按钮。

[0041] 存储器接口102可以与存储器150相耦合。该存储器150可以包括高速随机存取存储器和/或非易失性存储器,例如一个或多个磁盘存储设备,一个或多个光学存储设备,和/

或闪存存储器(例如NAND,NOR)。存储器150可以存储操作系统152,例如Android、iOS或是Windows Phone之类的操作系统。该操作系统152可以包括用于处理基本系统服务以及执行依赖于硬件的任务的指令。存储器150还可以存储应用154。在移动设备运行时,会从存储器150中加载操作系统152,并且由处理器104执行。应用154在运行时,也会从存储器150中加载,并由中央处理器104和/或图形处理器103执行。应用154运行在操作系统152之上,利用操作系统以及底层硬件提供的接口实现各种用户期望的功能,如即时通信、网页浏览、图片管理等。应用154可以是独立于操作系统152提供的,也可以是操作系统152自带的。另外,应用154被安装到移动终端100中时,也可以向操作系统152中添加驱动模块。

[0042] 在上述各种应用154中,其中的一种应用为根据本发明的基于卷积神经网络(Convolutional Neural Network,下文简称为CNN)的图像处理装置156。装置156实际上是一系列代码指令,其用于指示GPU103执行相应的图像处理操作。根据一种实施例,装置156由OpenGL ES图形程序接口来实现,即,装置156的代码需符合OpenGL ES的编码规则,从而提高编码效率、以及提高GPU的图像渲染效率。装置156中的CNN已经完成训练,适于接收待处理图像,将待处理图像作为输入,进行前向计算,从而完成对待处理图像的图像处理(例如图像分类、图像分割、图像风格转换、图像画质改善等),最终输出结果图像。

[0043] 图2示出了根据本发明一个实施例的CNN的结构图。如图2所示,本发明的CNN包括多个数据层(A)和多个处理层(B),更准确地说,图2中的CNN包括K个数据层和(K-1)的处理层。每个数据层包括多个特征图,一个特征图中包括多个数据点,每个数据点可以用来表示一个像素值。处理层用于对上一个数据层的多个特征图进行处理,以生成下一个数据层的多个特征图。处理层例如可以是卷积层、反卷积层、池化层、激活层等,但不限于此。待处理图像作为第一个数据层A1输入CNN,经过各处理层的处理,最终得到最后一个数据层AK,即结果图像,作为CNN的输出。

[0044] 图3示出了根据本发明一个实施例的图形处理器(GPU)103、中央处理器(CPU)104、以及存储器150的示意图,以更清楚地说明本发明的基于CNN的图像处理过程。

[0045] 如图3所示,GPU103包括着色器和图形存储器。着色器进一步包括顶点着色器和片段着色器,顶点着色器用于插值纹理的坐标位置,片段着色器用于对每个坐标位置进行相应的计算处理,顶点着色器和片段着色器相互配合,可以实现对图像的渲染处理,也即,可以实现CNN中各处理层的计算功能。图形存储器用于存储GPU计算所需要的数据。

[0046] GPU103和CPU104均可以从存储器150中读取数据,存储器150例如可以是内存。存储器150中存储有CNN的各项网络参数,例如每一个数据层的每一个特征图所对应的数值范围、每一个数据层的特征图的数量和尺寸、每一个数据层所对应的纹理图的尺寸以及其所包括的纹理区块的数量等。此外,存储器150中还存储有CNN各处理层所对应的OpenGL ES渲染脚本(即基于CNN的图像处理装置156),以及待处理图像中各像素的RGB值。

[0047] 传统方法通常采用CPU104来实现如图2所示的CNN的前向计算。由于CNN中涉及大量的浮点数计算,而CPU的浮点数计算效率较低,因此,采用CPU来执行CNN计算效率较低,无法满足移动终端的复杂图像处理需求。

[0048] 而对于本发明的基于CNN的图像处理方法来说,GPU103的图形存储器可用于存储CNN的网络参数(包括每一个数据层的每一个特征图所对应的数值范围、每一个数据层的特征图的数量和尺寸、每一个数据层所对应的纹理图的尺寸以及其所包括的纹理区块的数量

等)、纹理图(纹理图以第一数据类型存储有CNN当前数据层的特征图)以及基于OpenGL ES来编写的各处理层的渲染脚本(即装置156)。

[0049] 特别地,本发明的纹理图中以第一数据类型(uint8)存储数据,对于CNN的各处理层,先将纹理图中的数据转化为第二数据类型(float16或float32),再对第二数据类型的数据进行渲染处理。第一数据类型(uint8)可以满足大部分GPU芯片的存储空间限制,使得本方案可以适用于绝大多数的移动终端,兼容性较好。此外,纹理图中包括多个纹理区块,每个纹理区块包括RGBA四个通道,每个通道可以存储一张第一特征图。这样,所有的特征图数据均存储于一张纹理图中,在CNN的各处理层进行渲染处理时,从纹理图的各纹理区块的各通道中读取数据,进行分块渲染,无需创建多个纹理,避免了纹理渲染过程中不断地绑定、解绑纹理的操作,从而节省了计算时间,提高了计算效率。

[0050] 在执行本发明的基于CNN的图像处理方法时,CPU104向GPU103发出执行CNN计算的指示,GPU103接收该指示,将待处理图像中各像素的RGB值、CNN网络参数以及CNN渲染脚本从存储器150拷贝到图形存储器中。对于CNN的每一个处理层,着色器从图形存储器的纹理图中读取当前数据层的多个特征图、网络参数和渲染脚本,将各特征图中的数据转化为第二数据类型,再按照渲染脚本对第二数据类型的数据进行处理,得到下一个数据层的多个特征图,将下一个数据层的各特征图中的数据转化为第一数据类型并存储于纹理图中。当CNN的最后一个处理层完成计算后,即可得到结果图像。GPU103将结果图像输出至存储器150,并通知CPU104计算完成。在本发明的技术方案中,除了待处理图像的输入以及结果图像的输出,CNN各处理层的计算完全由GPU完成,而无需与CPU或存储器进行数据交换,充分利用了GPU并行浮点数计算的优势,大大提高了计算效率。

[0051] 图4示出了根据本发明一个实施例的基于CNN的图像处理方法400的流程图。在本发明中,CNN已经完成了训练,适于接收待处理图像,将待处理图像作为输入,进行前向计算,从而完成对待处理图像的图像处理(例如图像分类、图像分割、图像风格转换、图像画质改善等),最终输出结果图像。本发明的CNN包括多个数据层和多个处理层,每个数据层包括多个特征图。

[0052] 方法400适于在具有GPU的移动终端(例如前述移动终端100)中执行。GPU中包括图形存储器。GPU的图形存储器中存储有纹理图和网络参数,纹理图中以第一数据类型存储有当前数据层的多个第一特征图,网络参数包括每一个数据层的每一个特征图所对应的数值范围,每一个数据层的特征图的数量和尺寸,每一个数据层所对应的纹理图的尺寸和其所包括的纹理区块的数量,等等。应当指出,这些网络参数是事先确定的,在方法400的执行过程中,可以直接从图形存储器中取出网络参数使用,而无需重新计算。

[0053] 根据一种实施例,每一个数据层的每一个特征图所对应的数值范围按照以下步骤S1～S3来确定:

[0054] 步骤S1、将预定数量的测试图像输入CNN,进行前向计算。根据一种实施例,预定数量的测试图像包括第一数量的真实图像和第二数量的随机图像,其中,第一数量≥第二数量。随机图像的RGB值为采用高斯分布函数随机生成。高斯分布函数的均值μ和标准差σ亦是一定范围内的随机数,例如,若图片的RGB值采用0～255范围内的8位无符号整型(uint8)表示,则高斯分布的均值μ为0～255范围内的随机整数;而标准差σ则根据高斯分布的3σ原则,为满足不等式μ+3σ≤255的随机数。

[0055] 应当指出，为了使计算出来的数值范围更加准确，测试图像的数量应该足够多，即，第一数量和第二数量的值比较大，例如，二者均大于3000。特别地，如果参与计算的真实图像足够多，仅靠真实图像即可以得出较为准确的统计结果，则也可以将第二数量设置为0，即不采用随机图像。

[0056] 步骤S2、记录每一个测试图像在CNN的计算过程中所得到的每一个数据层的每一个特征图算例。特征图算例指的是测试图像在CNN的前向计算过程中所得到的特征图，例如，参考图2，对于测试图像fig1进行CNN的前向计算，数据层A1包括三个特征图，即测试图像fig1的RGB值，这三个特征图分别是一个特征图算例；经过处理层B1的计算，得到数据层A2，数据层A2的每一个特征图也都是一个特征图算例。

[0057] 步骤S3、对于CNN的每一个数据层的每一个特征图，根据该特征图所对应的所有测试图像的特征图算例来确定该特征图的数值范围。

[0058] 根据一种实施例，按照以下步骤S31、S32来确定一个特征图的数值范围：

[0059] 步骤S31、计算该特征图所对应的所有特征图算例的数据点均值的均值和标准差，特征图算例的数据点均值为特征图算例中各数据点的值的均值。

[0060] 例如，第k个数据层的第j个特征图的尺寸为a*b(1≤k≤CNN所包括的数据层的总数量K，1≤j≤第k个数据层所包括的特征图的总数量J(k))，即该特征图的横向包括a个数据点、纵向包括b个数据点，测试图像n(1≤n≤测试图像的总数量N)经过CNN前向计算可以得出该特征图的一个特征图算例$featuremap_{kj,n}$，则该特征图算例的数据点均值$\mu_{kj,n}$为：

[0061] $$\mu_{kj,n}=\frac{1}{a*b}\sum_{i=1,j=1}^{a,b}p(i,j) \qquad (1)$$

[0062] 其中，p(i,j)表示特征图算例$featuremap_{kj,n}$中坐标为(i,j)的数据点的值(或称像素值)。

[0063] 则第k个数据层的第j个特征图$featuremap_{kj}$所对应的所有特征图算例$featuremap_{kj,n}$(n＝1,2,…,N)的数据点均值的均值μ_{kj}为：

[0064] $$\mu_{kj}=\frac{1}{N}\sum_{n=1}^{N}\mu_{kj,n} \qquad (2)$$

[0065] 标准差σ_{kj}为：

[0066] $$\sigma_{kj}=\sqrt{\frac{1}{N}\sum_{n=1}^{N}(\mu_{kj,n}-\mu_{kj})^2} \qquad (3)$$

[0067] 应当指出，标准差也可以不采用上述式(3)来计算，而是先分别计算每一个特征图算例的数据点平方均值$\mu^2_{kj,n}$，再根据各特征图算例的数据点均值的均值μ_{kj}和数据点平方均值$\mu^2_{kj,n}$来计算。特征图算例$featuremap_{kj,n}$的数据点平方均值$\mu^2_{kj,n}$为：

[0068] $$\mu^2_{kj,n}=\frac{1}{a*b}\sum_{i=1,j=1}^{a,b}p^2(i,j) \qquad (4)$$

[0069] 其中，$p^2(i,j)$表示特征图算例$featuremap_{kj,n}$中坐标为(i,j)的数据点的值的平方。

[0070] 基于各特征图算例的数据点均值的均值μ_{kj}和数据点平方均值$\mu_{kj,n}^2$，第k个数据层的第j个特征图$featuremap_{kj}$所对应的所有特征图算例$featuremap_{kj,n}$(n＝1,2,…,N)的数据点均值的标准差σ_{kj}为：

[0071] $$\sigma_{kj}=\sqrt{(\frac{1}{N}\sum_{n=1}^{N}\mu_{kj,n}^2)-\mu_{kj}^2} \quad (5)$$

[0072] 步骤S32、根据步骤S31中计算出的均值和标准差来确定该特征图的数值范围。根据一种实施例，该特征图的数值范围设置为(min_{kj},max_{kj})，其中，$min_{kj}=\mu_{kj}-3\sigma_{kj}$，$max_{kj}=\mu_{kj}+3\sigma_{kj}$。

[0073] 如图4所示，方法400始于步骤S410。

[0074] 在步骤S410中，从纹理图中获取当前数据层的多个第一特征图。

[0075] 应当指出，此处的当前数据层(A_{now})不是特指CNN中的某一个数据层，而可以是CNN中除了待处理图像(第一个数据层A1)和结果图像(最后一个数据层AK)之外的任意一个数据层。类似地，下文中的当前处理层(B_{now})、下一个数据层(A_{next})也不是特指CNN中的某一个处理层或某一个数据层，而是相对于当前数据层而言。当前处理层、下一个数据层分别指的是与当前数据层相连的处理层、以及位于当前数据层之后的数据层。此外，此处的第一特征图、以及下文中的第二特征图中也不是特指某一个特征图，只是为了方便表述，而将当前数据层的特征图命名为第一特征图、将下一个数据层的特征图命名为第二特征图。例如，如图2所示，当前数据层A_{now}可以是数据层A2或数据层A3。若数据层A2为当前数据层，则当前处理层为处理层B2，下一个数据层为数据层A3，第一特征图指的是数据层A2中的特征图，第二特征图指的是数据层A3中的特征图。若数据层A3为当前数据层，则当前处理层为处理层B3，下一个数据层为数据层A4，第一特征图指的是数据层A3中的特征图，第二特征图指的是数据层A4中的特征图。

[0076] 随后，在步骤S420中，对于每一个第一特征图，根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型。根据一种实施例，第一数据类型为八位无符号整型(uint8)，以节省存储空间，兼容大部分GPU芯片；第二数据类型为浮点型(float16或float32)，以在渲染时进行浮点计算。

[0077] 根据一种实施例，可以按照以下步骤来将第一特征图中的数据转化为第二数据类型：首先，将第一特征图中的数据由0～255范围内的整数归一化为0.0～1.0范围内的浮点数，例如，可以将0～255范围内的整数均除以255，即可将其归一化为0.0～1.0范围内的浮点数。该步骤是由OpenGL ES的特性来决定的，OpenGL ES在渲染时，会自动执行该步骤，将0～255范围内的整数归一化为0.0～1.0范围内的浮点数。随后，将0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数，例如，可以按照以下公式0.0～1.0范围内的浮点数转化为该第一特征图所对应的数值范围内的浮点数：

[0078] $$f_{kj_1}=sf_{kj_1}\times(\max_{kj_1}-\min_{kj_1})+\min_{kj_1} \quad (6)$$

[0079] 其中，$(\min_{kj_1},\ \max_{kj_1})$为当前数据层k的第$j_1$个第一特征图的数值范围，$f_{kj1}$为$(\min_{kj_1},\ \max_{kj_1})$范围内的浮点数，$sf_{kj1}$为0.0～1.0范围内的浮点数。

[0080] 随后，在步骤S430中，当前处理层对第二数据类型的多个第一特征图进行渲染处

理，以生成下一个数据层的多个第二特征图。应当指出，当前处理层包括多种类型，例如卷积层、反卷积层、池化层、激活层，等等。根据一种实施例，渲染处理的步骤由OpenGL ES图形程序接口来完成，图形存储器中存储有适于执行渲染处理的OpenGL ES脚本。

[0081] 经过步骤S430的处理，可以得到下一个数据层的多个第二特征图，这时，第二特征图中的数据仍为第二数据类型，即float16或float32。

[0082] 随后，在步骤S440中，对于每一个第二特征图，根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型。从而方便实现第一数据类型与第二数据类型的转化，提高GPU的存储效率以及CNN的计算效率。

[0083] 根据一种实施例，可以按照以下步骤来将第二特征图中的数据转化为第一数据类型：首先，根据该第二特征图所对应的数值范围来将该第二特征图中的数据转化为0.0～255.0范围内的浮点数，例如，可以按照以下公式来将第二特征图中的数据转化为0.0～255.0范围内的浮点数：

[0084]

$$u_{(k+1)j_2}=\begin{cases}255.0, & f_{(k+1)j_2}>\max_{(k+1)j_2}\\ \dfrac{f_{(k+1)j_2}-\min_{(k+1)j_2}}{\max_{(k+1)j_2}-\min_{(k+1)j_2}}\times 255.0, & \min_{(k+1)j_2}\le f_{(k+1)j_2}\le \max_{(k+1)j_2}\\ 0.0, & f_{(k+1)j_2}<\min_{(k+1)j_2}\end{cases} \quad (7)$$

[0085] 其中，$(\min_{(k+1)j_2},\ \max_{(k+1)j_2})$下一个数据层(k+1)的第$j_2$个第二特征图的数值范围，$u_{(k+1)j_2}$为0.0～255.0范围内的浮点数，$f_{(k+1)j_2}$为下一个数据层(k+1)的第$j_2$个第二特征图中的任意一个数据点的值。随后，将0.0～255.0范围内的浮点数转化为小于等于其本身的最大整数。该步骤相当于将0.0～255.0范围内的浮点数(float)强制转换0～255范围内的整数(uint8)，从而将第二特征图中的数据转化为第一数据类型。

[0086] 随后，在步骤S450中，将多个第二特征图以第一数据类型存储至所述纹理图中。

[0087] 应当指出，此处的纹理图是一张大纹理图，该纹理图被划分为多个纹理区块，每个纹理区块包括RGBA四个通道，每个通道可以存储一个第二特征图。根据一种实施例，步骤S450相当于，将多个第二特征图以第一数据类型按顺序存储至各纹理区块的各通道中。

[0088] 根据一种实施例，图形存储器所存储的网络参数还包括第二特征图的数量和尺寸(第二特征图的尺寸包括第二特征图的横向数据点的数量和纵向数据点的数量)，以及下一个数据层所对应的纹理图所包括的纹理区块的数量以及该纹理图的尺寸。其中，纹理区块的数量根据第二特征图的数量来确定，纹理图的尺寸根据第二特征图的尺寸和纹理区块的数量来确定。例如，纹理区块的数量为ceil(c/4)，其中，c为第二特征图的数量，ceil(c/4)表示对(c/4)向上取整，即，取大于等于(c/4)的最小整数。纹理图的尺寸可以按照以下步骤确定：将纹理区块的数量因数分解为w*h，以使得(w*第二特征图的横向数据点的数量)与(h*第二特征图的纵向数据点的数量)的差值的绝对值最小；将纹理图的横向数据点的数量设置为(w*第二特征图的横向数据点的数量)，纵向数据点的数量设置为(h*第二特征图的纵向数据点的数量)。

[0089] 例如，经过前述步骤S410～S440，得到了23个第二特征图，每个特征图所包括的横

向数据点的数量为80，纵向数据点的数量为64，即，第二特征图的数量为23，尺寸为80*64。则，纹理区块的数量为ceil(23/4)＝6。将6因数分解为w*h，有四种分解方法：1)w＝1，h＝6；2)w＝2，h＝3；3)w＝3，h＝2；4)w＝6，h＝1。将第二特征图的横向数据点的数量记为a，纵向数据点的数量记为b，分别计算在上述每一种分解方法下，|w*a-h*b|的值：

[0090] 1) |w*a-h*b|＝|1*80-6*64|＝304；

[0091] 2) |w*a-h*b|＝|2*80-3*64|＝32；

[0092] 3) |w*a-h*b|＝|3*80-2*64|＝112；

[0093] 4) |w*a-h*b|＝|6*80-1*64|＝416；

[0094] 显然，在上述四种分解方法中，第2)种分解方法的|w*a-h*b|的值最小，因此，纹理图横向上有2个纹理区块，纵向上有3个纹理区块，纹理图的横向数据点的数量为2*80＝160，纵向数据点的数量为3*64＝192，也即，纹理图的尺寸为(2*80)*(3*64)＝160*192。

[0095] 按照上述方法设置纹理图的尺寸是因为，OpenGL ES支持的纹理的大小有限制，纹理的宽和高均需限制在2048个像素的范围内。上述方法可以使纹理图的横向数据点的数量与纵向数据点的数量最为接近，从而可以使得有限的空间内所存储的纹理数据量最大化。

[0096] 对将纹理图划分为多个纹理区块可以将多个第二特征图存储于一个纹理图中，在CNN的各处理层进行渲染处理时，从纹理图的各纹理区块的各通道中读取数据，进行分块渲染，无需创建多个纹理，避免了纹理渲染过程中不断地绑定、解绑纹理的操作，从而节省了计算时间，提高了计算效率。通过设置纹理图的尺寸，可以确定纹理区块的排列情况，使得纹理图中的数据存储得更为紧凑，提高存储空间的利用率。

[0097] 应当指出，以上详细介绍纹理区块的数量以及纹理图的尺寸的确定方法，只是为了更清楚地说明本发明的原理以及优点。但是，各数据层所对应的纹理图所包括的纹理区块的数量以及纹理图的尺寸并非在执行方法400的时候才确定，而是在执行方法400之前，作为CNN的网络参数预先存储于图形存储器中。在执行方法400时，可以直接从图形存储器中读取，无需重新计算。

[0098] 基于纹理区块的数量以及纹理图的尺寸(即纹理区块的排列情况)，可以将多个第二特征图存储于纹理图中。根据一种实施例，将第i个第二特征图以第一数据类型存储至第ceil(i/4)个纹理区块的第(mod(i-1,4)+1)个通道中，其中，mod(i-1,4)表示(i-1)除以4所得的余数。例如，经过前述步骤S410～S440，得到了23个第二特征图，每个特征图所包括的横向数据点的数量为80，纵向数据点的数量为64，即，第二特征图的数量为23，尺寸为80*64。相应地，下一个数据层所对应的纹理图中所包括的纹理区块的数量为6，纹理图的尺寸为(2*80)*(3*64)＝160*192。如图5所示，在23个第二特征图中，将第二特征图1存储至第1个纹理区块的第1个通道(R通道)中，将第二特征图2存储至第1个纹理区块的第2个通道(G通道)中，以此类推，将第二特征图23存储至第6个纹理区块的第3个通道(B通道)中，至此，所有23个第二特征图都完成了存储。这时，第6个纹理区块的第4个通道(A通道)未存储数据，基于补全原则，可以将第6个纹理区块的第4个通道的数据全部填充为0。

[0099] 经过步骤S450，下一个数据层(A_{next})的多个第二特征图以第一数据类型存储于纹理图中。接下来，可以以该数据层(A_{next})作为当前数据层(A_{now})，再次执行方法400，直至下一个数据层(A_{next})为倒数第二个数据层(A(K-1))为止。例如，如图2所示，在CNN中，数据层A2～数据层A3的计算过程可以用方法400来执行，数据层A2为当前数据层，数据层A3为下一

个数据层;随后,将数据层A3作为当前数据层,数据层A3～数据层A4的计算过程也可以用方法400来执行,得到下一个数据层A4;以此类推,直至将数据层A(K-2)作为当前数据层,得到下一个数据层A(K-1)。

[0100] 数据层A(K-1)的下一个数据层为数据层AK,即结果图像。数据层A(K-1)～数据层AK的计算过程有一定特殊性,其不一定适用方法400。若结果图像需要继续存储于GPU的图形存储器中以备他用,则数据层An可以按照步骤S450的方法存储于纹理图中,数据层A(K-1)～数据层AK的计算过程适用方法400。若结果图像不需要继续存储于GPU中,而是直接作为图像处理结果反馈给CPU(进一步可以展示给用户),则数据层AK的数据无需被限制于某个数值范围内、无需被转化为第一数据类型、也无需存储于纹理图中,不必执行步骤S440、S450,因此,这种情况下,数据层A(K-1)～数据层AK的计算过程不再适用方法400。

[0101] 根据一种实施例,在结果图像直接作为图像处理结果反馈给CPU的情况下,数据层A(K-1)～数据层AK的计算过程如下:从纹理图中获取倒数第二个数据层(A(K-1))的多个特征图;对于倒数第二个数据层的每一个特征图,根据其所对应的数值范围来将该特征图中的数据转化为第二数据类型;CNN的最后一个处理层(B(K-1))对倒数第二个数据层的多个特征图进行渲染处理,以生成结果图像;将结果图像输出至存储器中进行存储。同时,GPU向CPU发出图像处理完成的指令,CPU可以从存储器中读取结果图像以展示给用户或做他用。

[0102] 此外,应当指出,数据层A1～数据层A2的计算过程也有一定的特殊性。因为数据层A1为待处理图像,此时,纹理图中尚未存储数据。此外,待处理图像的RGB数据本身为浮点型(第二数据类型),可以直接进行CNN计算,因此,也完全没有必要先将待处理图像的RGB数据转化为第一数据类型(uint8)存储于纹理图中,再从纹理图中取出第一数据类型的数据并将其转化为第二数据类型进行计算。因此,数据层A1～数据层A2的计算过程不必执行步骤S410、S420,不适用方法400。

[0103] 根据一种实施例,数据层A1～数据层A2的计算过程如下:将存储器中的待处理图像缓存至图形存储器中,并将待处理图像作为第一个数据层(A1)输入到CNN;CNN的第一个处理层(B1)对待处理图像进行渲染处理,以生成第二个数据层(A2)的多个特征图;对于第二个数据层的每一个特征图,根据其所对应的数值范围来将该特征图中的数据转化为第一数据类型;将第二个数据层的多个特征图以第一数据类型存储至所述纹理图中。

[0104] 根据本发明的技术方案,GPU的图形存储器中存储有纹理图,纹理图中以第一数据类型(uint8)存储数据,对于CNN的各处理层,先将纹理图中的数据转化为第二数据类型(float16或float32),再对第二数据类型的数据进行渲染处理。第一数据类型(uint8)可以满足大部分GPU芯片的存储空间限制,使得本方案可以适用于绝大多数的移动终端,兼容性较好。

[0105] 本发明采用一张纹理图来存储当前数据层的多个第一特征图,纹理图中包括多个纹理区块,每个纹理区块包括RGBA四个通道,每个通道可以存储一张第一特征图。这样,所有的特征图数据均存储于一张纹理图中,在CNN的各处理层进行渲染处理时,从纹理图的各纹理区块的各通道中读取数据,进行分块渲染,无需创建多个纹理,避免了纹理渲染过程中不断地绑定、解绑纹理的操作,从而节省了计算时间,提高了计算效率。

[0106] 由于本发明的纹理图中存储了当前数据层的所有特征图,且图形存储器中存储有CNN的各项网络参数以及OpenGL ES脚本,在GPU执行CNN计算时,可以直接从图形存储器中

快速读写数据并进行渲染。除了待处理图像的输入以及结果图像的输出，各处理层的计算完全由GPU完成，而无需与CPU、内存进行数据交换，充分利用了GPU并行浮点数计算的优势，大大提高了计算效率，使得移动端的复杂图像处理(例如图像风格转换、视频风格转换等)成为可能。

[0107] 经过测试，本发明的优化后GPU图像处理方案与传统的CPU图像处理方案相比，计算效率可以提高8～10倍。

[0108] A9：A8所述的方法，其中，所述网络参数还包括第二特征图的数量和尺寸，以及下一个数据层所对应的纹理图所包括的纹理区块的数量和纹理图的尺寸，其中，所述第二特征图的尺寸包括第二特征图的横向数据点的数量和纵向数据点的数量；

[0109] 所述纹理区块的数量为ceil(c/4)，其中，c为第二特征图的数量，ceil(c/4)表示大于等于(c/4)的最小整数；

[0110] 所述纹理图的尺寸按照以下步骤确定：

[0111] 将纹理区块的数量因数分解为w*h，以使得(w*第二特征图的横向数据点的数量)与(h*第二特征图的纵向数据点的数量)的差值的绝对值最小；

[0112] 所述纹理图的横向数据点的数量为(w*第二特征图的横向数据点的数量)，所述纹理图的纵向数据点的数量为(h*第二特征图的纵向数据点的数量)。

[0113] A10：A9所述的方法，其中，所述将多个第二特征图以第一数据类型按顺序存储至各纹理区块的各通道中的步骤包括：将第i个第二特征图以第一数据类型存储至第ceil(i/4)个纹理区块的第(mod(i-1,4)+1)个通道中，其中，mod(i-1,4)表示(i-1)除以4所得的余数。

[0114] A11：A1-10中任一项所述的方法，其中，所述移动终端还包括存储器，所述存储器中存储有待处理图像；所述方法还包括：

[0115] 将存储器中的所述待处理图像缓存至所述图形存储器中，并将所述待处理图像作为第一个数据层输入到所述卷积神经网络；

[0116] 所述卷积神经网络的第一个处理层对所述待处理图像进行渲染处理，以生成第二个数据层的多个特征图；

[0117] 对于第二个数据层的每一个特征图，根据其所对应的数值范围来将该特征图中的数据转化为第一数据类型；

[0118] 将第二个数据层的多个特征图以第一数据类型存储至所述纹理图中。

[0119] A12：A11所述的方法，其中，还包括：

[0120] 从所述纹理图中获取倒数第二个数据层的多个特征图；

[0121] 对于倒数第二个数据层的每一个特征图，根据其所对应的数值范围来将该特征图中的数据转化为第二数据类型；

[0122] 所述卷积神经网络的最后一个处理层对倒数第二个数据层的多个特征图进行渲染处理，以生成结果图像；

[0123] 将所述结果图像输出至所述存储器中进行存储。

[0124] A13：A1-12中任一项所述的方法，其中，所述渲染处理的步骤由OpenGL ES图形程序接口来完成，所述图形存储器中还存储有适于执行所述渲染处理的OpenGL ES脚本。

[0125] 这里描述的各种技术可结合硬件或软件，或者它们的组合一起实现。从而，本发明

的方法和设备，或者本发明的方法和设备的某些方面或部分可采取嵌入有形媒介，例如可移动硬盘、U盘、软盘、CD-ROM或者其它任意机器可读的存储介质中的程序代码(即指令)的形式，其中当程序被载入诸如计算机之类的机器，并被所述机器执行时，所述机器变成实践本发明的设备。

[0126] 在程序代码在可编程计算机上执行的情况下，计算设备一般包括处理器、处理器可读的存储介质(包括易失性和非易失性存储器和/或存储元件)，至少一个输入装置，和至少一个输出装置。其中，存储器被配置用于存储程序代码；处理器被配置用于根据该存储器中存储的所述程序代码中的指令，执行本发明的基于卷积神经网络的图像处理方法。

[0127] 以示例而非限制的方式，可读介质包括可读存储介质和通信介质。可读存储介质存储诸如计算机可读指令、数据结构、程序模块或其它数据等信息。通信介质一般以诸如载波或其它传输机制等已调制数据信号来体现计算机可读指令、数据结构、程序模块或其它数据，并且包括任何信息传递介质。以上的任一种的组合也包括在可读介质的范围之内。

[0128] 在此处所提供的说明书中，算法和显示不与任何特定计算机、虚拟系统或者其它设备固有相关。各种通用系统也可以与本发明的示例一起使用。根据上面的描述，构造这类系统所要求的结构是显而易见的。此外，本发明也不针对任何特定编程语言。应当明白，可以利用各种编程语言实现在此描述的本发明的内容，并且上面对特定语言所做的描述是为了披露本发明的最佳实施方式。

[0129] 在此处所提供的说明书中，说明了大量具体细节。然而，能够理解，本发明的实施例可以在没有这些具体细节的情况下被实践。在一些实例中，并未详细示出公知的方法、结构和技术，以便不模糊对本说明书的理解。

[0130] 类似地，应当理解，为了精简本公开并帮助理解各个发明方面中的一个或多个，在上面对本发明的示例性实施例的描述中，本发明的各个特征有时被一起分组到单个实施例、图、或者对其的描述中。然而，并不应将该公开的方法解释成反映如下意图：即所要求保护的本发明要求比在每个权利要求中所明确记载的特征更多特征。更确切地说，如下面的权利要求书所反映的那样，发明方面在于少于前面公开的单个实施例的所有特征。因此，遵循具体实施方式的权利要求书由此明确地并入该具体实施方式，其中每个权利要求本身都作为本发明的单独实施例。

[0131] 本领域那些技术人员应当理解在本文所公开的示例中的设备的模块或单元或组件可以布置在如该实施例中所描述的设备中，或者可替换地可以定位在与该示例中的设备不同的一个或多个设备中。前述示例中的模块可以组合为一个模块或者此外可以分成多个子模块。

[0132] 本领域那些技术人员可以理解，可以对实施例中的设备中的模块进行自适应性地改变并且把它们设置在与该实施例不同的一个或多个设备中。可以把实施例中的模块或单元或组件组合成一个模块或单元或组件，以及此外可以把它们分成多个子模块或子单元或子组件。除了这样的特征和/或过程或者单元中的至少一些是相互排斥之外，可以采用任何组合对本说明书(包括伴随的权利要求、摘要和附图)中公开的所有特征以及如此公开的任何方法或者设备的所有过程或单元进行组合。除非另外明确陈述，本说明书(包括伴随的权利要求、摘要和附图)中公开的每个特征可以由提供相同、等同或相似目的的替代特征来代替。

[0133] 此外，本领域的技术人员能够理解，尽管在此所述的一些实施例包括其它实施例中所包括的某些特征而不是其它特征，但是不同实施例的特征的组合意味着处于本发明的范围之内并且形成不同的实施例。例如，在下面的权利要求书中，所要求保护的实施例的任意之一都可以以任意的组合方式来使用。

[0134] 此外，所述实施例中的一些在此被描述成可以由计算机系统的处理器或者由执行所述功能的其它装置实施的方法或方法元素的组合。因此，具有用于实施所述方法或方法元素的必要指令的处理器形成用于实施该方法或方法元素的装置。此外，装置实施例的在此所述的元素是如下装置的例子：该装置用于实施由为了实施该发明的目的的元素所执行的功能。

[0135] 如在此所使用的那样，除非另行规定，使用序数词“第一”、“第二”、“第三”等等来描述普通对象仅仅表示涉及类似对象的不同实例，并且并不意图暗示这样被描述的对象必须具有时间上、空间上、排序方面或者以任意其它方式的给定顺序。

[0136] 尽管根据有限数量的实施例描述了本发明，但是受益于上面的描述，本技术领域内的技术人员明白，在由此描述的本发明的范围内，可以设想其它实施例。此外，应当注意，本说明书中使用的语言主要是为了可读性和教导的目的而选择的，而不是为了解释或者限定本发明的主题而选择的。因此，在不偏离所附权利要求书的范围和精神的情况下，对于本技术领域的普通技术人员来说许多修改和变更都是显而易见的。对于本发明的范围，对本发明所做的公开是说明性的而非限制性的，本发明的范围由所附权利要求书限定。

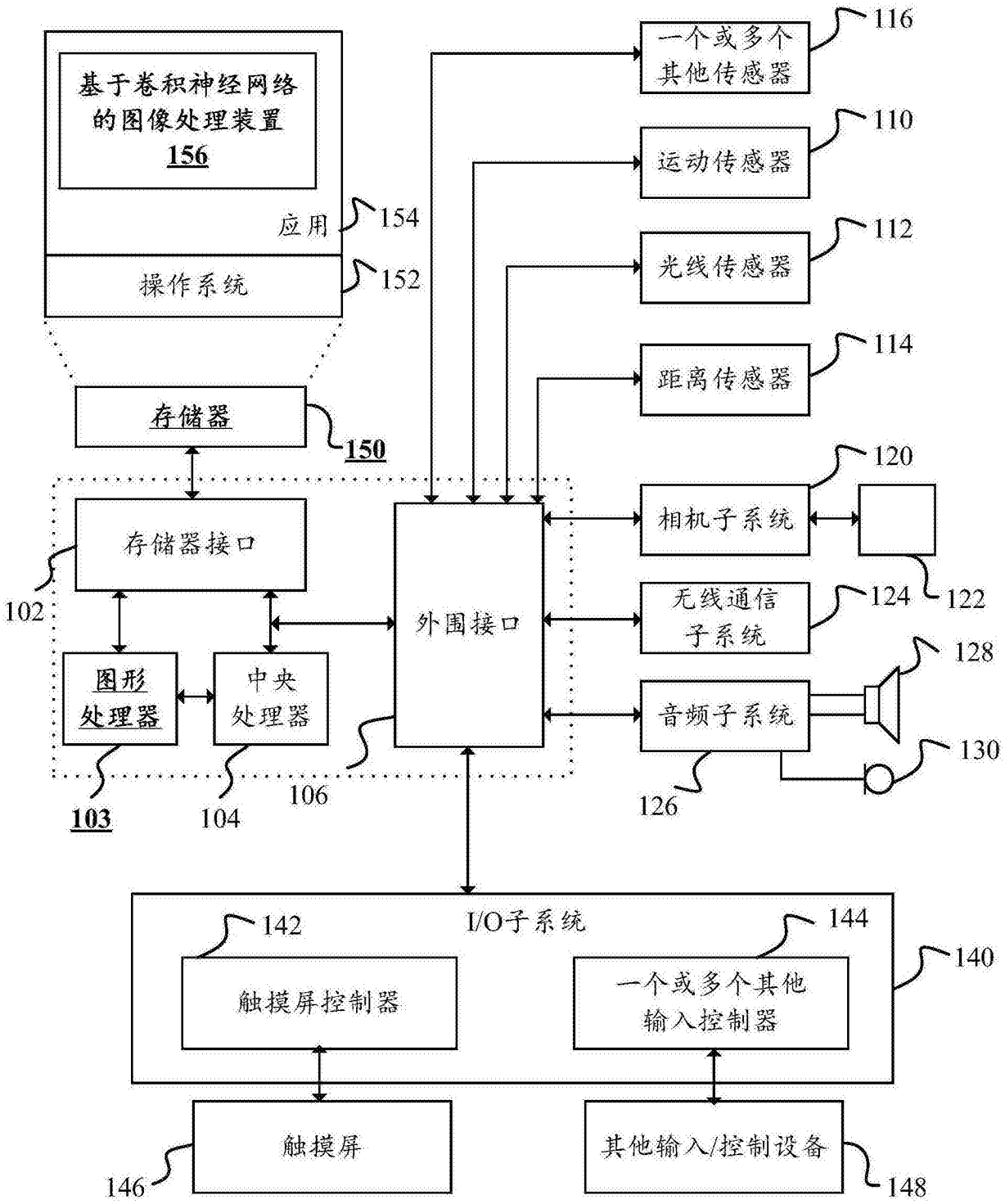

基于卷积神经网络的图像处理装置
156
应用
154
操作系统
152
存储器
150
存储器接口
102
图形处理器
103
中央处理器
104
外围接口
106
一个或多个其他传感器
116
运动传感器
110
光线传感器
112
距离传感器
114
相机子系统
120
122
无线通信子系统
124
音频子系统
126
128
130
I/O子系统
140
触摸屏控制器
142
一个或多个其他输入控制器
144
触摸屏
146
其他输入/控制设备
148

图1

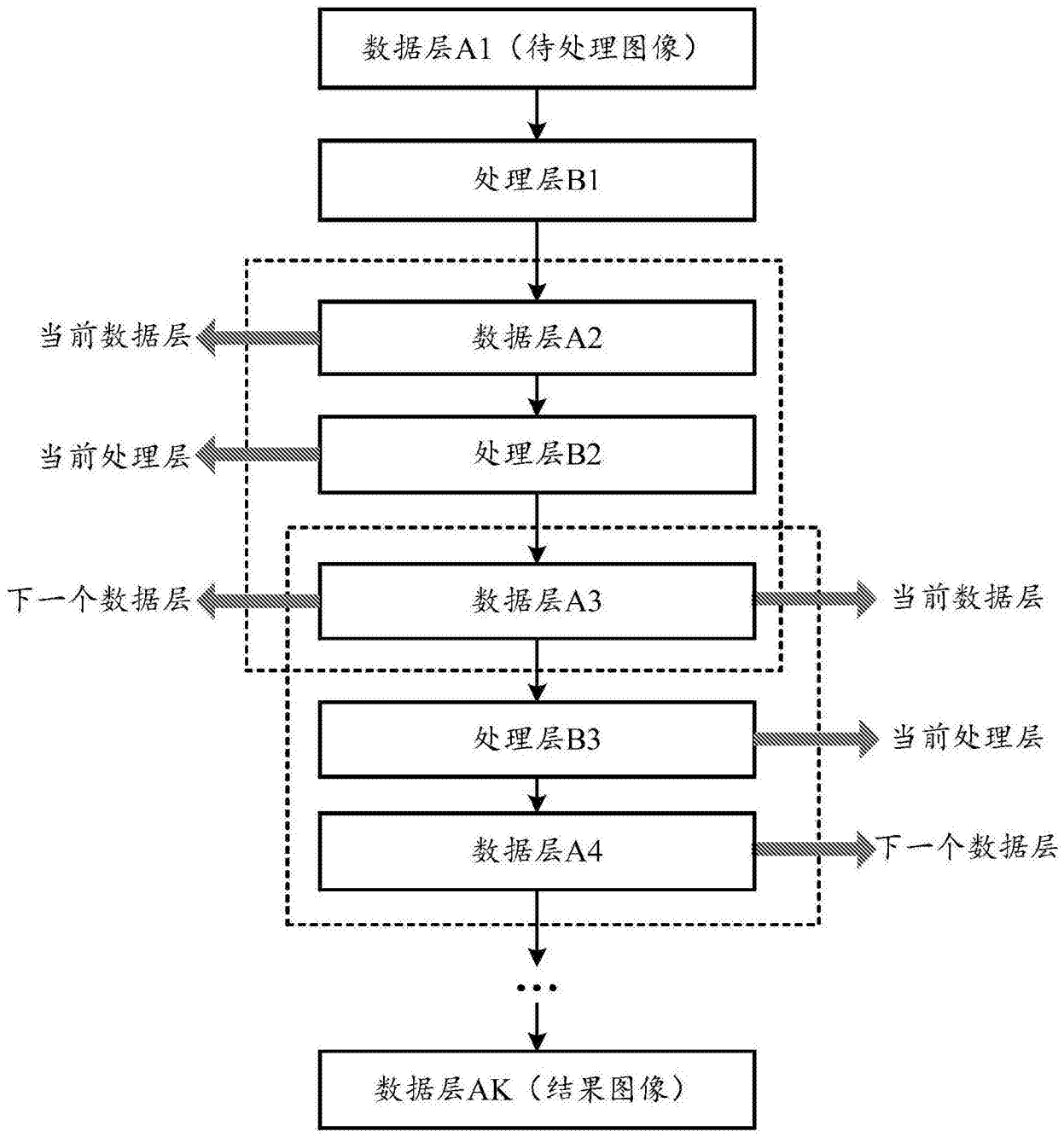

数据层A1（待处理图像）
处理层B1
当前数据层
数据层A2
当前处理层
处理层B2
下一个数据层
数据层A3
当前数据层
处理层B3
当前处理层
数据层A4
下一个数据层
数据层AK（结果图像）

图2

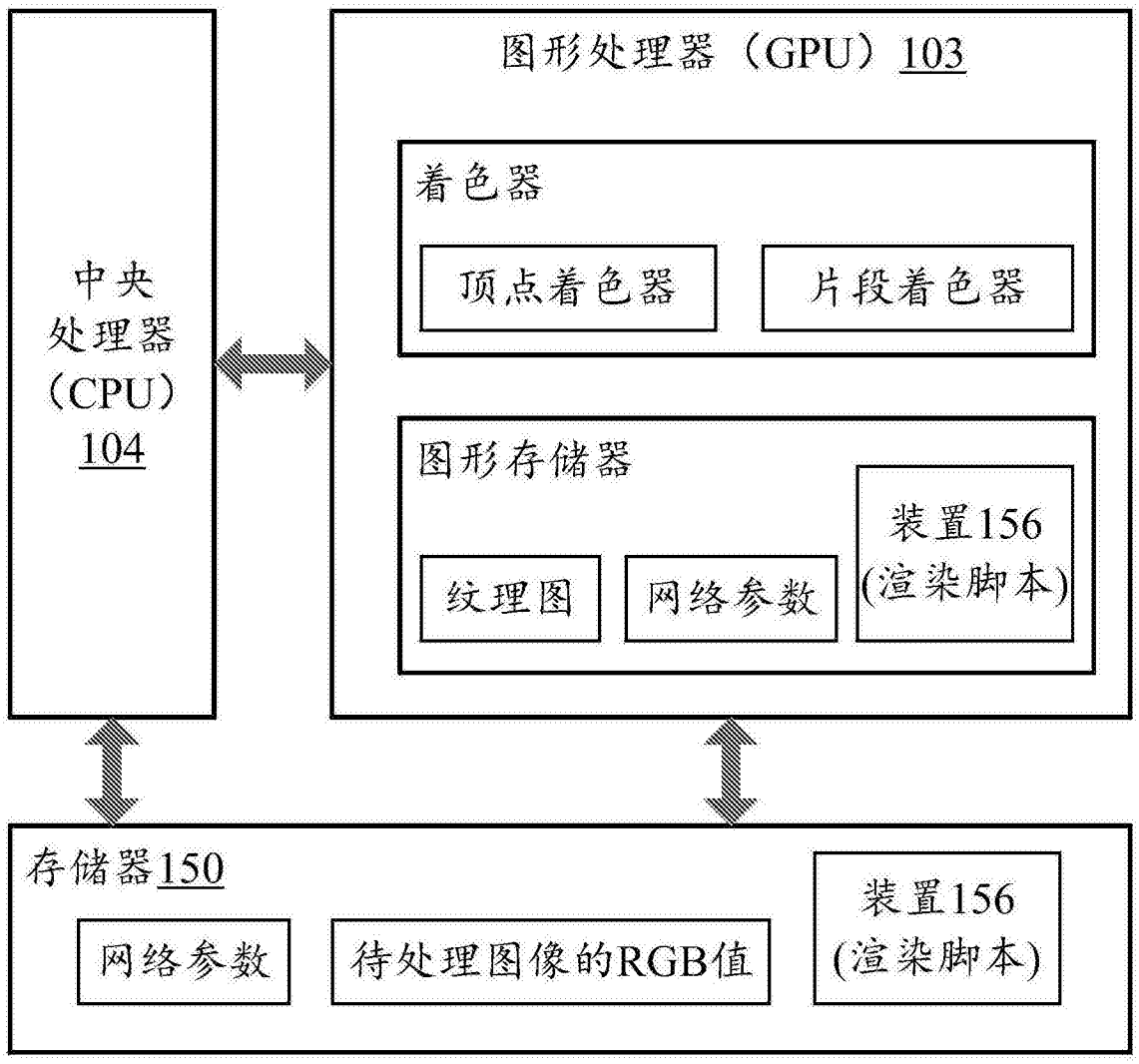

图形处理器（GPU）103
着色器
顶点着色器
片段着色器
图形存储器
纹理图
网络参数
装置156
(渲染脚本)
中央
处理器
(CPU)
104
存储器150
网络参数
待处理图像的RGB值
装置156
(渲染脚本)

图3

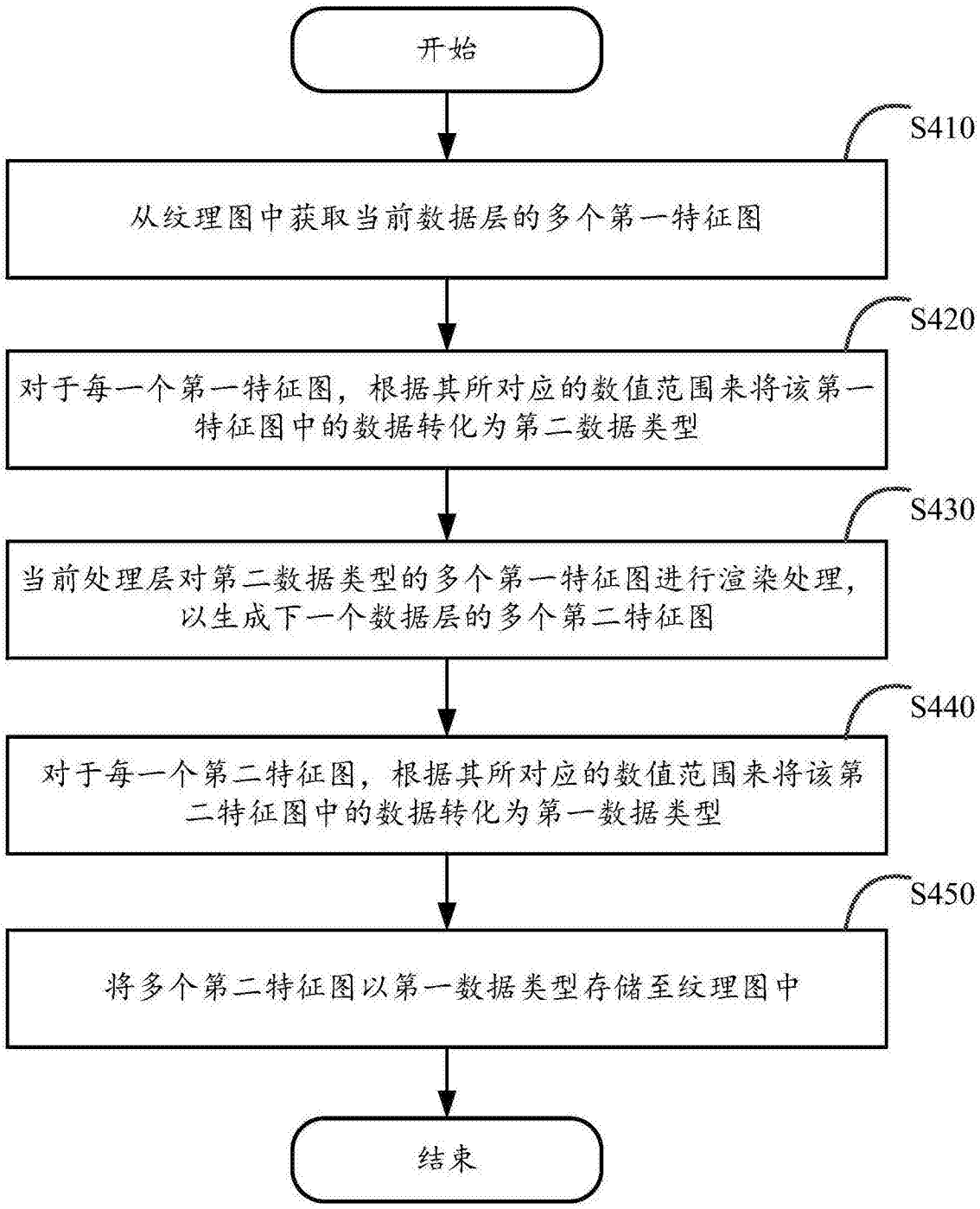

开始
S410
从纹理图中获取当前数据层的多个第一特征图
S420
对于每一个第一特征图，根据其所对应的数值范围来将该第一特征图中的数据转化为第二数据类型
S430
当前处理层对第二数据类型的多个第一特征图进行渲染处理，以生成下一个数据层的多个第二特征图
S440
对于每一个第二特征图，根据其所对应的数值范围来将该第二特征图中的数据转化为第一数据类型
S450
将多个第二特征图以第一数据类型存储至纹理图中
结束

图4

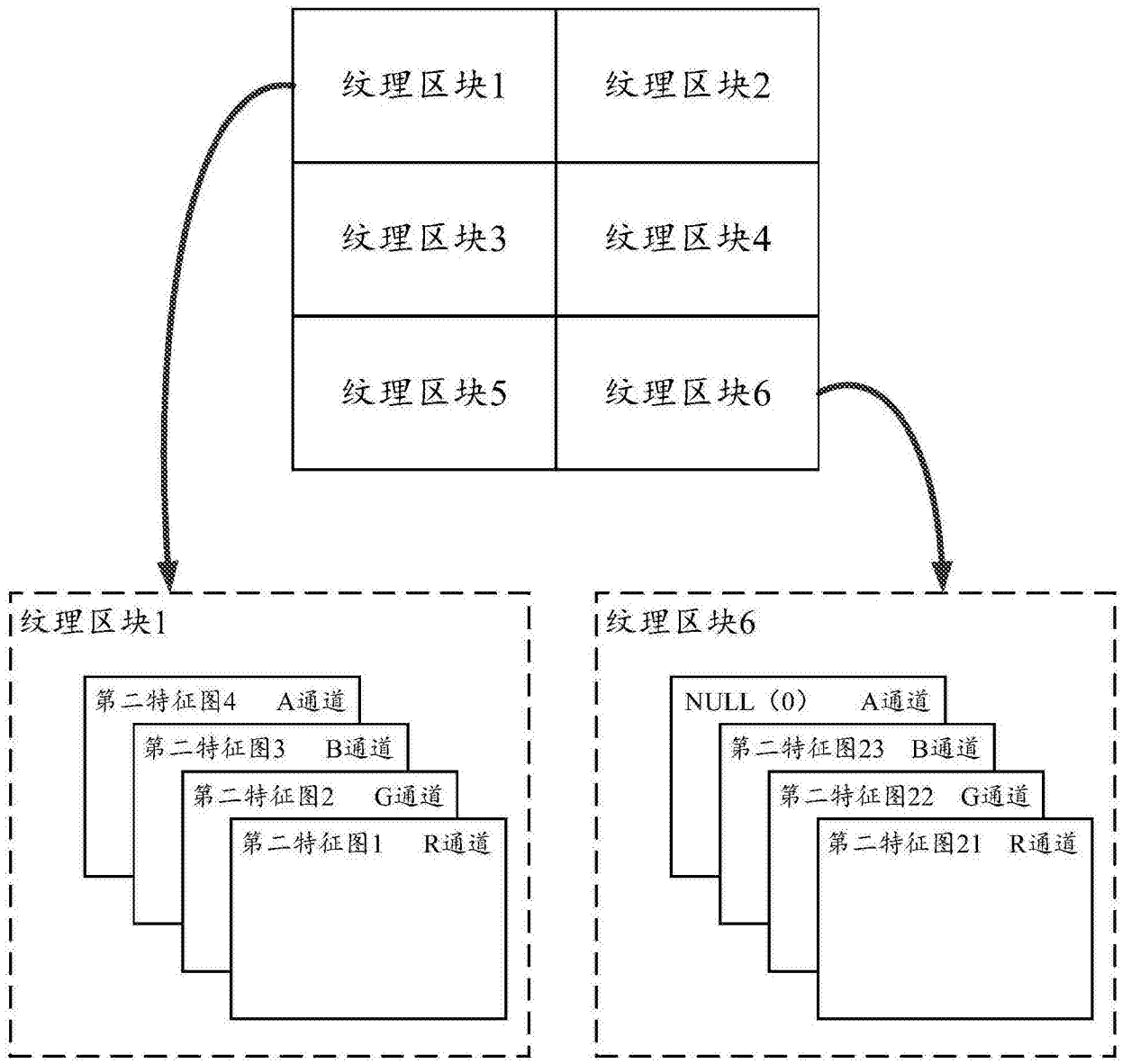

纹理区块1
纹理区块2
纹理区块3
纹理区块4
纹理区块5
纹理区块6
纹理区块1
第二特征图4 A通道
第二特征图3 B通道
第二特征图2 G通道
第二特征图1 R通道
纹理区块6
NULL（0） A通道
第二特征图23 B通道
第二特征图22 G通道
第二特征图21 R通道

图5