一种应用于图形显示的 Upsampling IP

团队介绍

参赛单位：南京大学

队伍名称：啊对对对对对对

指导老师：沙金

参赛队员：朱涛、徐振华、李子倞

1.背景

在GPU图形处理领域，像素性能常常是系统性能的瓶颈，减轻像素引擎的负载对于提升芯片图形处理能力有着重大意义。渲染低分辨率后，经由上采样算法将图片放大至期望大小已经成为一个被广泛使用的性能优化手段,从NVIDIA的DLSS到AMD的FSR都有广泛的应用。

2.算法设计介绍

基于以上背景，针对给定的1k图片集，我们提出了多种将1k图片放大至4k的插值算法，其中不同的算法实现了性能与复杂度之间不同的权衡。对于传统双线性插值放大，4k图像会直接利用原1k图像的像素点，另一方面，放大4倍时传统双线性插值会导致整体图像的偏移，进而影响性能。因此，我们改变了4k像素点位置，使1k图片与4k图片中心对齐，得到新型双线性插值算法。由于给定的1k图片集是经均值下采样获得，原4k图像16点的平均值等于相应位置1k图像的像素值，在还原的4k图像中，我们也希望保持这一性质，因此提出了均值调整这一步骤。

对于梯度变化剧烈的边缘区域，我们又提出了一种利用1k图片像素点进行的边缘插值算法，现以求解水平方向第一行像素点为例，如图1所示。我们首先利用线性插值求出第一行左右两端的像素值pixel\_left和pixel\_right，而我们所求4点位于，当所求像素的横坐标小于跳变点x1，其像素值赋值为pixel\_left，当大于x1，赋值为pixel\_right。同理，我们也可以求出水平方向其余3行以及竖直方向4列的像素值，两者加权取平均即为最终的边缘插值算法。可以看出，边缘插值算法过分的拉大了边缘的梯度，中间不存在过渡区，为缓解这一问题，我们还需在生成4k图片的基础上对4k图片进行的高斯平滑。



图 1 边缘插值示意图

针对上述的新型双线性插值、均值调整、边缘插值、高斯平滑，以及最近邻插值，我们给出了以下不同组合算法，其中组合4为最终作为硬件实现的算法。

表 1 各种组合算法

|  |  |
| --- | --- |
| 组合1 | 新型双线性插值+均值调整 |
| 组合2 | 新型双线性插值+均值调整+高斯平滑+均值调整 |
| 组合3 | 分区域插值+均值调整+高斯平滑+均值调整 |
| 组合4 | 边缘插值+均值调整+高斯平滑+均值调整 |
| 组合5 | 边缘插值+均值调整+高斯平滑 |

其中分区域插值指的是对于1k图片像素点，我们卷积上Sobel算子以便求出水平与竖直梯度变化，当两者均小于某一阈值则采用最近邻插值，当两者中有一个大于某一阈值则采用边缘插值，其余区域采用新型双线性插值。

表 2 各种组合算法性能

|  |  |  |  |
| --- | --- | --- | --- |
|  | 平均PSNR | 平均SSIM | 平均LPIPS |
| 官方参考 | 28.36 | 0.800 | 0.284 |
| 组合1 | 31.73 | 0.871 | 0.212 |
| 组合2 | 31.88 | 0.873 | 0.223 |
| 组合3 | 32.11 | 0.875 | 0.211 |
| 组合4 | 32.08 | 0.874 | 0.205 |
| 组合5 | 31.97 | 0.870 | 0.224 |

对于以上各种组合算法，可以看出，不同组合方式具有不同的性能和算法复杂度。其中，组合1复杂度最低且具有相对高的性能，组合3复杂度最高，PSNR与SSIM性能也最优，而组合4在3个区域内均采用边缘插值，相比于组合3复杂度有所下降，而性能基本不变。因此，我们选取组合4作为最终硬件实现的算法。在经过四舍五入后，组合4的平均PSNR=32.08，SSIM=0.874，LPIPS=0.205，相比于官方给的PSNR=28.36，SSIM=0.800，LPIPS=0.284均有着较大幅度的提升，且算法复杂度不高，具有广阔的应用前景。

3.硬件设计介绍

我们在Xilinx Zynq-7020 FPGA开发板上搭建了整个系统，下图2为硬件设计的系统架构。其总体流程为首先PS端将SD卡内的1k图片写入到DDR3，当存储完毕后写入1k存储完成标志位，而PL端通过AXI\_HP0接口一直访问1k图片存储完成标志位，若存储完成，PL端开始从DDR3读取1k像素数据，并将经RGB三颜色插值模块生成的4k像素数据写回DDR3。当4k图片数据存储完毕时，PL端会向DDR3中写入4k图片存储完成标志位，而PS端会一直读取4k存储完成标志位的值，若满足要求，则PS端开始从SD卡中读取下一张1k图片，循环往复，直至完成所有图片的插值。

当所有图片插值完成后，PS端会通过AXI\_GP0接口实现对VDMA的配置与初始化，并将DDR3中的4k数据通过AXI\_HP1接口输入到VDMA，进而将像素数据输出到外接显示屏上。当我们在PS端对VDMA进行重新配置时，即可改变VDMA的显示内容，进而实现分时显示4k图片的功能。



图 2 系统硬件框架图

其中axi\_ddr\_rw模块用来控制对DDR3内数据的存与取，以下图3是该模块实现的功能状态机。



图 3 axi\_ddr\_rw功能状态机

对于单颜色分量插值模块而言，其功能为接收1k图片一列的5个单颜色分量数据，并利用先前存储的1k像素，实现边缘插值+均值调整+高斯平滑+均值调整的功能，进而求得最终4k图片的单个颜色分量的像素值，单颜色分量插值模块设计如图4所示：



图 4 单颜色分量插值模块框架

在进行边缘插值+均值调整的过程中，由于我们要对3个1k像素点进行边缘插值+均值调整，每个点需要花费1个时钟，从流水线的角度看，整个插值模块需要花费3个时钟才能求出一个1k像素点对应的16个4k像素点，在100M的时钟下，若仅考虑理想状态，每张图片需要的时钟数为：



因此，1s钟能够处理的图片数为帧。然而，由于我们需要从DDR3中取出1k数据以及存储4k数据，而我们采用的DDR3数据位宽为32bit，每个时钟仅能存取一个像素点，因此，实际处理速率会受到很大的影响，其可以通过异步fifo来提高DDR3的存取时钟来解决，本设计中不再讨论。

4.硬件资源消耗

图5为顶层模块所耗资源示意图，其使用LUT数为11220，使用FF数为3586，可以看出，整个算法复杂度相对较低，所用资源也较少。

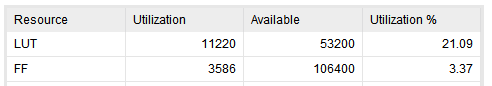


图 5 顶层资源消耗

图6为各模块LUT的资源消耗，新型双线性插值模块使用LUT数为837，其复杂度与传统双线性插值一致；边缘插值模块使用LUT数为1815，近似为新型双线性插值模块的2.2倍；16点均值调整模块使用LUT数为605；单点高斯平滑使用LUT数为40，若对16点进行高斯平滑，其使用总LUT数为。

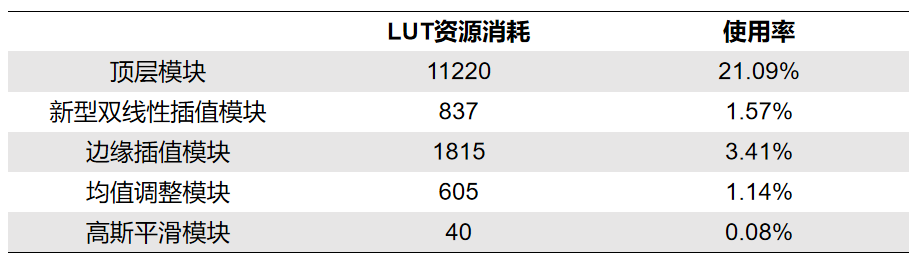


图 6 各模块LUT资源消耗

5.结果展示

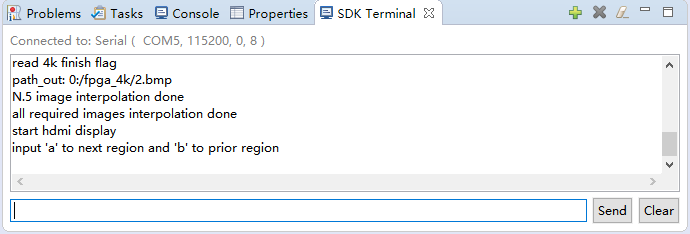


图 7 串口打印进程信息

|  |  |
| --- | --- |
| C:\Users\master\Desktop\final\比赛素材\图3.jpg | C:\Users\master\Desktop\final\比赛素材\图4.jpg |
| C:\Users\master\Desktop\final\比赛素材\图1.jpg | C:\Users\master\Desktop\final\比赛素材\图2.jpg |

图 8 改变显示内容效果图

图7展现串口打印的进程信息，此时所有图片已经插值完成，PS端成功配置VDMA并开始HDMI显示，此时我们可以在SDK Terminal输入字符a以及字符b来改变显示内容，其中a表示下一区域，b表示上一区域。图8展现了通过输入字符来改变显示内容对应外接显示屏的显示效果，与预期结果一致。

6.参赛体会

在参与景嘉微杯的过程中，我们学习了关于图像超分辨率的若干算法，同时，在给定均值下采样后，提出了一些较为新颖的图像插值算法，并成功在全可编程FPGA Zynq-7020上搭建了整个系统，结果表明，我们提出的算法在性能以及硬件复杂度上均取得了较为优异的效果。

比赛已到达尾声，在此，由衷感谢大赛组委会为我们提供了集创赛这一优质平台，让我们能够与全国各地的优秀学子同台切磋，共同进步，也感谢组委会们在大赛中的辛勤付出，最后，预祝集创赛能越办越好。