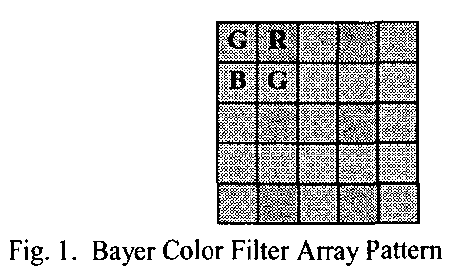
Ocl\_dcraw文档

# 简介

数码相机感光器件通常只能取得RGB中的一种数值，通过合理排列加上适当算法就能还原出所有像素的RGB值。例如：



不同的排列方法，构成不同的感光器模板。常用如图2\*2的bayer模板。感光值矩阵和其他附加信息构成raw格式。

Dcraw是将数码相机raw格式图像转换成ppm或tiff格式的开源程序。dcraw使用纯C语言，可以在任何系统上运行。Ocl\_dcraw是利用opencl和显卡的计算能力对dcraw在效率上的改进。目前ocl\_dcraw只对dcraw的一个分支流程做了优化。该流程调用的主要函数有：nikon\_compressed\_load\_raw, scale\_colors, pre\_interpolate, ahd\_interpolate, border\_interpolate, convert\_to\_rgb. 下面分别介绍每个函数的功能和在ocl\_dcraw的改进。

# nikon\_compressed\_load\_raw

nikon\_compressed\_load\_raw是对nikon的压缩图像进行解压，输出为一个表示感光值的矩阵。压缩格式类似jpeg, 利用huffman码表示长度，记录数值间的插值。具体算法请参考源代码。Dcraw对内存的使用非常小心。为降低内存消耗，dcraw没有为外部文件开设任何buffer。 Ocl\_dcraw更注重效率，因此为外部文件开设了buffer，以减少IO的访问次数。

# scale\_colors

scale\_colors根据不同的白平衡策略，调整感光值矩阵数值比例。Ocl\_dcraw实现scale\_colors的gpu版本。Kernel中每个线程计算一个2\*2的像素快。具体算法比较简单，可直接参考代码。

# pre\_interpolate

pre\_interpolate对不同的感光器模板，确定感光值与RGBA的关系，并调整到相应的位置。最新的ocl\_dcraw将pre\_interpolate和scale\_colors合并，在一个kernel中实现，以降低访存时间。具体算法比较简单，可直接参考代码。

# ahd\_interpolate

ahd是一种差分算法。一共对图像进行三次扫描计算。假设图像为尺寸为n\*m. 第一次，计算n\*m内所有RB像素的G值。数据相关为(中心点为计算的像素，下同)：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

第二次扫描，计算所有RB值，由于要利用第一次扫描计算的结果，本次计算针对(n-2)\*(m-2)内像素。数据相关为：

如果该像素为G：

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

否则为：

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

第三次扫描计算CIELab,针对(n-4)\*(m-4)内像素，数据相关为：

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

最后(n-6)\*(m-6)内的像素值为最终结果存储。

Ocl\_dcraw将三次扫描计算分为三个kernel实现，分别对应每一次扫描。由于第一次扫描只针对RB像素点，第一个kernel中，每个线程计算一个2\*2的菱形像素块，如图：

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

这种方法在边界为锯齿型，因此第二个kernel针对(n-4)\*(m-4)内图像。第二，第三个kernel中，每个线程处理一个2\*2的方形像素块，如图：

|  |  |
| --- | --- |
|  |  |
|  |  |

# Border\_interpolate

如上节所述，ahd插分算法产生宽度为6的边界（在ocl\_dcraw中调整宽度为8），border\_interpolate对ahd产生的边界进行填充。Border\_interpolate算法较简单，结果取相邻的相应颜色值的平均。Ocl\_dcraw中，每个线程处理一个2\*2的方形像素块。

# convert\_to\_rgb

convert\_to\_rgb将感光值乘以相应系数转化为RGB值，以及统计颜色分布直方图。Ocl\_dcraw中每个线程对应一个像素点。

注：Ocl\_dcraw统计颜色直方图时，为提高效率没有使用元操作函数。因此直方图结果可能与cpu计算结果不同。由于直方图只用来调整亮度，饱和度等，对图像不会产生大的影响。若使用-w –W 参数时，可以省略计算直方图。

# 图像分块并行

读文件，解码和插分三个 过程可以进行流水线并行。为保持与cpu结果的一致，分块数据间有交叉部分。因此需要在原有基础上做些修改，具体方法请参看代码。