How to ADD lens shading correction in DNG file

1. Use opcodelist

Opcode list is a series of operations which can be applied to the raw image when Photoshop open the DNG file. Not all the processing software support this tag in DNG file.

As I tested, acdsee, faststone, they don’t support this function.

This feature is new after DNG version 1.3.

Now the latest version is 1.4.

1. Gainmap

Gainmap is one operation in opcodelist, which can applied gain to the image in raw domain.

1. Steps to do lens shading correction in DNG
2. Add opcodelist2 in sub ifd0

('tag',51009,'description','Opcodelist2','type',7,'count',4+4\*(16+76+4\*32\*32),'values',M.offset\_Opcodelist);

Size of the value for this DE is related to lens shading correction table, in Apical, it’s 32x 32, and float type, so for r,gr,gb,b colour plains, total size of the DE content is 4+4\*(16+76+4\*32\*32), M.offset\_Opcodelist store the offset of the DE content in file.

The most special thing in this tag is, the tag type is 7 which means undefined and one byte size, so in tag writing part of sub IFD, this must be treated specially, or the offset value can’t be write correctly. The offset should be written in 4 bytes not 1 bytes.

if (typ ==7)

opcode\_offset\_data = 2172040;

put\_value\_g(entry\_offset+8, opcode\_offset\_data, 4, byte\_order);

else

in write\_ifd\_entry\_g.m

1. Then from the offset place M.offset\_Opcodelist, there saved number of opcode which equal to 4 because for colour plains need 4 gain map, and 4 gain maps follow that.
2. The gain map parameters explanation.

opcodelist = {...

struct('description','NumberOfOpcodeGainMaps','type',4,'count',1,'values',[4]),...

struct('description','OpcodeheaderB.OpcodeIDGainMap','type',4,'count',1,'values',[9]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeSpecVer','type',4,'count',1,'values',[16973824]),... %16973824 means DNG 1.3.0.0

struct('description','opcodelist.OpcodeheaderB.OpcodeFlags','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeIDGainMapSize','type',4,'count',1,'values',[76+4\*32\*32]),... % size of each gainmap

% R channel:

struct('description','opcodelist.GainMapB.Top','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.Left','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Right','type',4,'count',1,'values',M.height),... % wrong in spec, this is width in spec, but accually,it’s height.

struct('description','opcodelist.GainMapB.Bottom','type',4,'count',1,'values',M.width),... % wrong in spec, this is height in spec, but accually,it’s width.

struct('description','opcodelist.GainMapB.Plane','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Planes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.RowPitch','type',4,'count',1,'values',[2]),... % means every 2 pixel is processed in row,such as 1, 1+2, 1+2+2,…

struct('description','opcodelist.GainMapB.ColPitch','type',4,'count',1,'values',[2]),... % means every 2 pixel is processed in colomn, such as 1, 1+2, 1+2+2,…

struct('description','opcodelist.GainMapB.MapPointsH','type',4,'count',1,'values',[32]),...% same as lsc table width, 32 in apical.

struct('description','opcodelist.GainMapB.MapPointsV','type',4,'count',1,'values',[32]),... % same as lsc table width, 32 in apical.

struct('description','opcodelist.GainMapB.MapSpacingH','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapSpacingV','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapOriginH','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapOriginV','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapPlanes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainB','type',11,'count',32\*32,'values',M.lensshading\_b),...

%Gr channel. struct('description','OpcodeheaderB.OpcodeIDGainMap','type',4,'count',1,'values',[9]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeSpecVer','type',4,'count',1,'values',[16973824]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeFlags','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeIDGainMapSize','type',4,'count',1,'values',[76+4\*32\*32]),...

struct('description','opcodelist.GainMapB.Top','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.Left','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.Right','type',4,'count',1,'values',M.height),...

struct('description','opcodelist.GainMapB.Bottom','type',4,'count',1,'values',M.width),...

struct('description','opcodelist.GainMapB.Plane','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Planes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.RowPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.ColPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.MapPointsH','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapPointsV','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapSpacingH','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapSpacingV','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapOriginH','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapOriginV','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapPlanes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainB','type',11,'count',32\*32,'values',M.lensshading\_g),...

%Gb channel. struct('description','OpcodeheaderB.OpcodeIDGainMap','type',4,'count',1,'values',[9]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeSpecVer','type',4,'count',1,'values',[16973824]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeFlags','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeIDGainMapSize','type',4,'count',1,'values',[76+4\*32\*32]),...

struct('description','opcodelist.GainMapB.Top','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Left','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Right','type',4,'count',1,'values',M.height),...

struct('description','opcodelist.GainMapB.Bottom','type',4,'count',1,'values',M.width),...

struct('description','opcodelist.GainMapB.Plane','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Planes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.RowPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.ColPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.MapPointsH','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapPointsV','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapSpacingH','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapSpacingV','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapOriginH','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapOriginV','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapPlanes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainB','type',11,'count',32\*32,'values',M.lensshading\_g),...

%B channel. struct('description','OpcodeheaderB.OpcodeIDGainMap','type',4,'count',1,'values',[9]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeSpecVer','type',4,'count',1,'values',[16973824]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeFlags','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.OpcodeheaderB.OpcodeIDGainMapSize','type',4,'count',1,'values',[76+4\*32\*32]),...

struct('description','opcodelist.GainMapB.Top','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Left','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.Right','type',4,'count',1,'values',M.height),...

struct('description','opcodelist.GainMapB.Bottom','type',4,'count',1,'values',M.width),...

struct('description','opcodelist.GainMapB.Plane','type',4,'count',1,'values',[0]),...

struct('description','opcodelist.GainMapB.Planes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainMapB.RowPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.ColPitch','type',4,'count',1,'values',[2]),...

struct('description','opcodelist.GainMapB.MapPointsH','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapPointsV','type',4,'count',1,'values',[32]),...

struct('description','opcodelist.GainMapB.MapSpacingH','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapSpacingV','type',12,'count',1,'values',[1.0/32]),...

struct('description','opcodelist.GainMapB.MapOriginH','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapOriginV','type',12,'count',1,'values',[0.0]),...

struct('description','opcodelist.GainMapB.MapPlanes','type',4,'count',1,'values',[1]),...

struct('description','opcodelist.GainB','type',11,'count',32\*32,'values',M.lensshading\_r)};

1. If written the converter in matlab.

Type 11 float and type 12 double of a tag should be treated specially.

Matlab default type is double, but the basic unit in DNG is byte, so write in byte with correct endian order should be considered when implemented.

This is how to write float type to dng file

function put4gf(offset, value, byte\_order)

% function data = put4(data, offset, value, byte\_order)

fid=fopen('temp','W+');

if byte\_order == 'b'

fwrite(fid,value,'single','b');

else

fwrite(fid,value,'single');

end

fclose(fid);

fid=fopen('temp','r');

b=fread(fid,4,'uchar');

fclose(fid);

global data;

for i=1:4

data(offset+i) = b(i);

end