

# DigiVFX

## HW1 HDR Imaging

B02901072 楊喬諳  
B02901065 李洺曦

### Description

我們整個程式都是使用MATLAB所撰寫  
使用到的程式庫只包含MATLAB所內建的hdrwrite( )函式用以儲存hdr file  
以及教授所介紹的Debevec's gsolve.m

程式碼包括

— myHDR.m

使用 `hdr = myHDR(ImageDir, format, algorithm, arg);`

ImageDir 為圖片檔資料夾位置

format 為圖片格式，如'.jpg'，'.png'，'.tif'等

algorithm可為1或2

若為1，則使用Debevec's method

arg 為exposure time的vector

若為2，則使用Mitsunaga and Nayar's method

hdr 回傳一組 圖片長x圖片寬x3的 hdr radiance matrix

會在程式碼中呼叫到gsolve.m gsolve2.m myMTB.m函式

— gsolve.m

Debevec's method 同講義中所介紹的一樣

— gsolve2.m

使用 `g = gsolve2.m(Z,M);`

Mitsunaga and Nayar's method 假設為M次多項式並解之

#reference : <High Dynamic Range Imaging> p.175開始

$$c_N = 1 - \sum_{n=0}^{N-1} c_n$$

The final  $N \times N$  system can be written:

$$\begin{bmatrix} \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,0} (d_{p,q,0} - d_{p,q,N}) & \dots & \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,0} (d_{p,q,N-1} - d_{p,q,N}) \\ \dots & \dots & \dots \\ \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,N-1} (d_{p,q,0} - d_{p,q,N}) & \dots & \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,N-1} (d_{p,q,N-1} - d_{p,q,N}) \end{bmatrix}$$
$$\times \begin{bmatrix} c_0 \\ \dots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} - \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,0} d_{p,q,N} \\ \dots \\ - \sum_{q=1}^{Q-1} \sum_{p=1}^P d_{p,q,N-1} d_{p,q,N} \end{bmatrix}$$

where

$$d_{p,q,n} = M_{p,q}^n - R_{q,q+1} M_{p,q+1}^n$$

— myMTB.m

使用  $X = \text{myMTB}(X);$

輸入一個 圖片數x圖片長x圖片寬x3 的 pixel value matrix

回傳一個 已aligned的 pixel value matrix

— myTonemap.m

我們使用的是講義上的 Photographic Tone Reproduction

使用方式有兩種

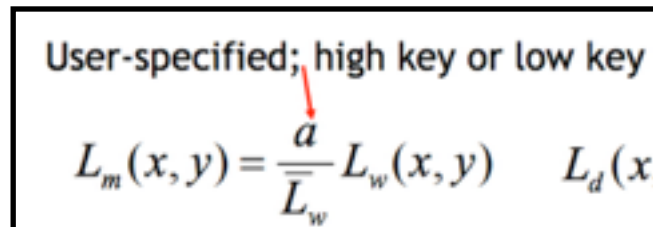
(1)  $\text{rgb} = \text{myTonemap}(\text{hdr}, 'global', \text{key}, L_{\text{white}});$

hdr 為一組 圖片長x圖片寬x3的 hdr radiance matrix

可以是myHDR所回傳的值，或是內建的hdrread所回傳的值

rgb 為tonemap出來的結果 可用imshow(rgb)加以顯示

key 為下圖中所使用的"a"

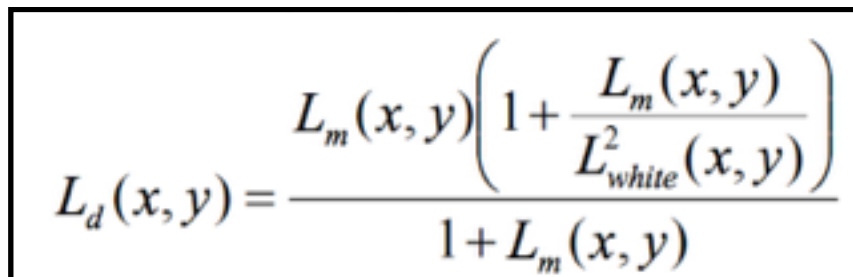


User-specified; high key or low key

$$L_d(x, y) = \frac{a}{L_w} L_m(x, y) \quad L_d(x, y)$$

$L_{\text{white}}$  為下圖中所使用 " $L_{\text{white}}$ "，可填可不填

default值為整張圖片中原有最高的  $L_m$  值


$$L_d(x, y) = \frac{L_m(x, y) \left( 1 + \frac{L_m(x, y)}{L_{\text{white}}^2(x, y)} \right)}{1 + L_m(x, y)}$$

## Image alignment -> myNTB.m

我們使用的是講義中所介紹的MTB alignment technique

先找到一張圖的中位數median，將threshold設為此，以上為1，以下為0  
然後再進行八個方向（加上原本不動共九個）可能進行微調，得出變化最小的  
先將圖片縮小到1/32，九個方向進行比較，然後是1/16, 1/8...以加快速度  
實作中，我們已曝光量中間程度那張（假設有九張，就是用第五張）作為標準圖  
將其他影像依據標準圖做移動



上圖無使用MTB，下圖有。可看出下圖的圖片經過align之後較上圖清晰許多。

以下使用我們Artifact所使用的照片為範例：



曝光時間由短至長分別為

0.125, 0.25, 0.4, 0.6, 0.8, 1.3, 2.0, 3.0, 5.0

光圈設為f-5.0

## **High Dynamic Range -> myHDR.m**

(a) Debevic's method -> gsolve.m

(b) Mitsunaga and Nayar's method -> gsolve2.m

## **Ghost Removal -> myGhost.m & myImageVariance.m**

這部分我們有嘗試做做看，但只有做到Image的Weighted Variance這一步而已

## **Tone Mapping -> myTonemap.m**