High Dynamic Range
Plage dynamique élevée

1. High Dynamic Range

Etant donné un appareil photo numérique, la dynamique du capteur correspond à la plage de variation entre le niveau minimal (niveau de noir) que le capteur peut percevoir et le niveau maximal avant saturation. La technique HDR permet de combiner plusieurs images de la même scène et prises du même point de vue afin d'obtenir une image ayant une dynamique très étendue.

1.1. Capteur et photosites et dynamique

Un appareil photo numérique est composé d'optiques, d'un capteur et d'une unité de traitement d'images. Le capteur est constitué d'un ensemble de photosites permettant de transformer la lumière incidente en charge électrique, dont l'intensité est linéaire en fonction de l'éclairement. Ce courant électrique est ensuite mesuré puis converti en un signal numérique qui est à l'origine, après une série de traitements, de l'intensité d'un pixel.

La sensibilité d'un photosite correspond au nombre minimal de photons incidents par unité de surface nécessaires pour générer un signal autre que du bruit. Dans le noir, un photosite génère un signal (le courant d'obscurité) dû à l'agitation thermique du capteur, dont le niveau moyen défini le niveau de noir.

Par ailleurs, au-delà d'un certain niveau d'éclairage, les photosites saturent. La dynamique du capteur correspond à la plage de variation entre le niveau minimal que le capteur peut percevoir (niveau de noir) et le niveau maximal avant saturation.

Ainsi, une image numérique est fidèle à la scène photographiée si l'amplitude entre l'éclairement le plus faible et le plus fort est inférieure à la dynamique de l'image. En pratique, cela n'est généralement pas le cas et il faut alors faire un choix lors de la prise de vue.

1.2. High Dynamic Range

Les techniques de High Dynamic Range (HDR), consistent à combiner plusieurs images de la même scène prises du même point de vue, mais avec des temps de poses différents afin de générer une image ayant une dynamique bien supérieure à celle des images initiales.

Sur l'exemple suivant, il est possible de prendre une photo avec un joli ciel bleu mais une herbe complétement saturée, ou bien l'inverse. L'image de droite correspond à l'image HDR construite à partir des 2 images de gauche.

2. Méthode proposée

Il existe diverses méthodes permettant de générer des images HDR. D'ailleurs, la plupart des smartphones proposent l'option HDR dans le menu de l'appareil photo. La méthode que nous allons voir n'est pas la plus populaire, mais une des plus performantes, elle présente par ailleurs l'intérêt de ne pas être trop compliquée à coder. Pour plus de détails sur cette méthode, vous pouvez lire l'article

Paul Debevec et Jitendra Malik : « Recovering High Dynamic Range Radiance Maps from Photographs », publié dans lors de la conférence Siggraph en 1997.

2.1. Courbe de réponse du capteur

La réponse électrique X d'un photosite est linéaire en fonction du temps d'exposition Δt. On a alors :

$$X = E\Delta t \ (J/m^2)$$

où E correspond à l'irradiance, c'est-à-dire à la quantité de lumière par unité de temps arrivant sur le photosite. Durant la phase de numérisation du signal électrique, celui-ci est traité par une fonction non-linéaire f propre à chaque appareil numérique. Ainsi, la valeur Z du pixel associé à un photosite vaut :

$$Z = f(X) = f(E\Delta t)$$

Le but de la méthode proposée ici est de calculer la courbe de réponse du capteur afin de trouver pour chaque pixel l'exposition X du photosite :

$$X = f^{-1}(Z)$$

puis d'en déduire l'irradiance de chaque pixel :

$$E = \frac{X}{\Delta t}$$

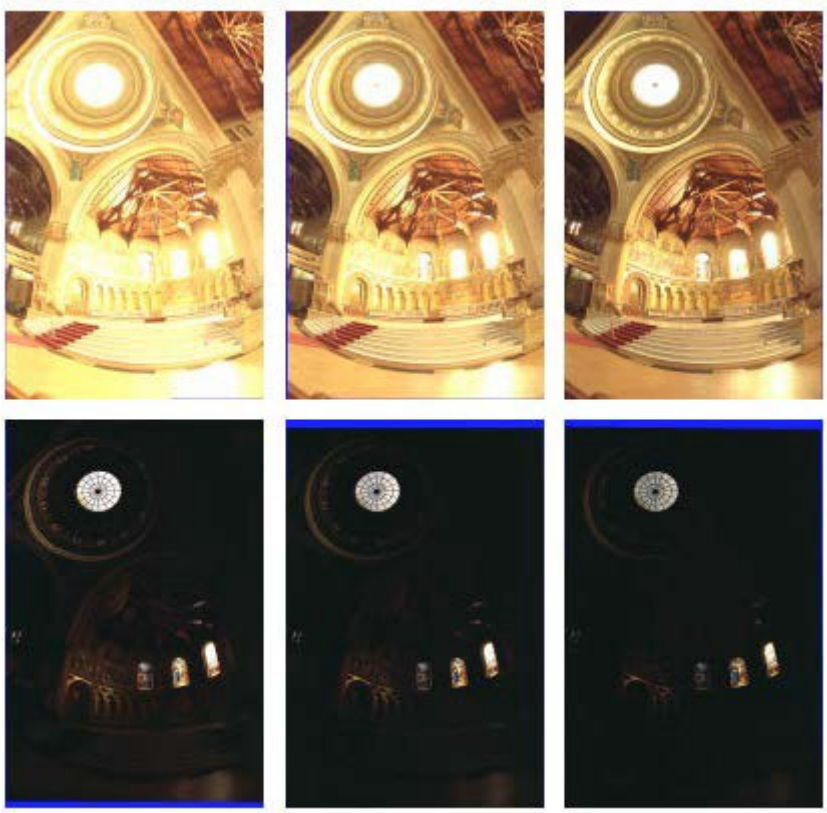
Pour la suite, nous considérons que les contraintes suivantes sont satisfaites :

- la scène considérée est statique durant la prise de vue (quelques secondes), aussi bien géométriquement qu'en terme de luminosité.
- la fonction f est monotone et croissante, ainsi f^{-1} est bien dénie.
- la fonction f est relativement lisse.

Par ailleurs, nous traiterons les différents canaux RGB séparément.

2.2. Inversion de la courbe de réponse

La méthode présentée ici nécessite en entrée une série d'images numériques de la même scène, prise du même point de vue, mais avec différents temps d'exposition.



Différents temps d'exposition : 1/800 s, 1/40 s, 0.5 s, 1.6 s, 5 s, and 15 s peuvent aboutir aux résultats illustrés ci-contre

Plus précisément, nous considérons :

- un ensemble de P images.
- chaque image est constituée de $N = w \times h$ pixels
- l'intensité du i-ème pixel ($i \in [1, N]$) de la j-ème image ($j \in [1, P]$) est notée $Z_{i,j}$.
- le temps d'exposition de la j-ème images est noté Δt_i .

Nous avons alors la relation de numérisation suivante :

$$Z_{i,j} = f(E_i \Delta t_j)$$

La fonction f étant inversible, nous avons :

$$E_i \, \Delta t_i = f^{-1}(Z_{i,j})$$

Soit

$$lnE_i + ln\Delta t_j = ln\, f^{-1}(Z_{i,j})$$

Par souci de simplicité d'écriture, nous renommons $\ln f^{-1}$ en g :

$$g(Z_{i,j}) = ln\, f^{-1}(Z_{i,j}) = \ln E_i + \ln \Delta t_j$$

En pratique, nous connaissons les $Z_{i,j}$ ainsi que les Δt_j , et nous cherchons

à définir la fonction $g(Z)$ ainsi que les E_i.

La fonction $g(Z)$ est définie sur l'intervalle des Z entiers $\in [Z_{min}; Z_{max}]$

(avec $Z_{min} = 0$ et $Z_{max} = 255$). Par ailleurs, les E_i sont identiques pour chaque prise de vue.

La méthode proposée consiste à trouver la fonction $g(Z)$ et les E_i en minimisant le résidu suivant :

$$O = \sum_{i=1}^{N}\sum_{j=1}^{P}\left(g(Z_{i,j}) - \ln E_i - \ln \Delta t_j\right)^2 + \lambda \sum_{Z=Z_{min}+1}^{Z_{max}-1} g^{"}(Z)^2$$

terme de fidélité aux données terme de lissage

avec λ une variable expérimentale ($\lambda = 20$ est un bon départ pour vos tests).

Par ailleurs, une bonne estimation de la dérivée seconde de g peut s'obtenir avec

$g^{"}(z) \approx g(z-1) - 2g(z) + g(z+1)$.

En réfléchissant très fort, il est possible d'écrire cette équation comme un système surdéterminé dont les inconnues sont les $Z_{max} - Z_{min} + 1 = 256$ valeurs de g ainsi que les $\ln E_i$. Le but de ce projet consiste en partie à trouver ce système linéaire $Ax = b$, le coder et le résoudre au sens des moindres carrés : $x = (A^T A)^{-1} A^T b$. Un indice, ignorez les carrés dans la formule du résidu, ils sont implicitement pris en compte dans les moindres carrés.

Par ailleurs, notez que les deux éléments du résidu contribuent chacun à des équations différentes d'un même système linéaire.

2.3. Facteur d'échelle

La solution du système surdéterminé précédent est défini à un facteur d'échelle prés. En effet, changer $\ln E_i$ en $\ln E_i + \alpha$ et $g(z)$ en $g(z) + \alpha$ ne change pas la solution du système. Pour stabiliser le système, nous ajoutons donc une nouvelle contrainte :

$$g(Z_{mid}) = 0$$

avec :

$$Z_{mid} = Z_{min} + Z_{max}$$

Cette contrainte peut s'ajouter facilement dans le système d'équations précédent.

2.4. Précision des données

Par expérience, on connait la forme de la courbe de réponse et notamment son caractère de pente raide auprès de Zmin et Zmax. Pour obtenir une meilleure adéquation aux données, nous introduisons une fonction de poids w(z) qui diminue l'influence des données en bordure de courbe :

$$w(Z) = \begin{cases} \mathrm{Z} - \mathrm{Zmin\ si\ Z} \leq \dfrac{\mathrm{Zmin} + \mathrm{Zmax}}{2} \\ \mathrm{Zmax} - \mathrm{Z\ si\ Z} > \dfrac{\mathrm{Zmin} + \mathrm{Zmax}}{2} \end{cases}$$

Le système à minimiser s'exprime alors :

$$O = \sum_{i=1}^{N}\sum_{j=1}^{P}\left(w(Z_{i,j})g(Z_{i,j}) - \ln E_i - \ln \Delta t_j\right)^2 + \lambda \sum_{Z=Z_{min}+1}^{Z_{max}-1}\left(w(Z)g^{"}(Z)\right)^2$$

2.5. Robustesse et temps de calcul

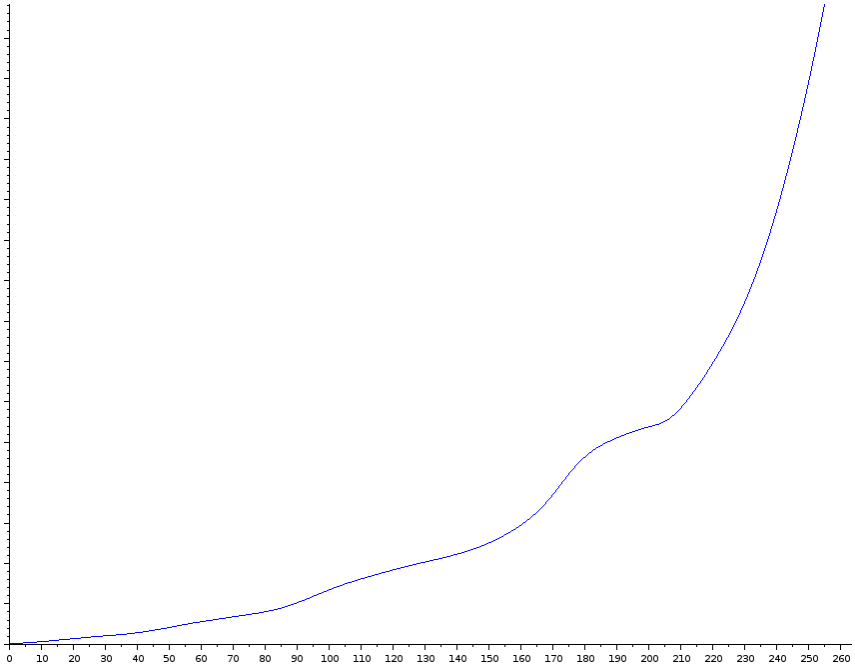
Pour que le système puisse avoir au moins une solution, il faut :

$$N(P-1) > Z_{max} - Z_{min}$$

Utiliser tous les pixels pour résoudre ce système serait beaucoup trop long, un petit nombre d'entre eux sera suffisant pour obtenir un résultat satisfaisant. Si l'on dispose de 6 photos, alors une centaine de pixels seront largement suffisants. L'idéal étant de les choisir tels que les E_i couvrent une large gamme d'illumination avec une bonne distribution entre les Z_{min} et les Z_{max}.

2.6. Au final

A l'issue de cette étape, la solution du système linéaire vous donne l'ensemble des valeurs de $g(Z)$ sur $Z \in [Z_{min}, Z_{max}]$ ainsi que les valeurs de E pour les pixels sélectionnés. Il reste donc à retrouver les valeurs E pour tous les pixels.



$e^{g(Z)}$

3. Reconstruction

3.1. Irradiance

Une fois la courbe g retrouvée, il faut calculer l'image d'irradiance, c'est-à-dire notre image HDR. Le mieux étant de la coder sur plus que 8 bit, des float ou des double feront l'affaire.

Pour retrouver les Ei, nous pouvons utiliser la relation suivante :

$$\ln E_i = g(Z_{i,j}) - \ln \Delta t_j$$

Pour obtenir une bonne précision numérique, il parait judicieux d'utiliser toutes les images et de faire, pour chaque pixel, la moyenne des irradiances trouvées. Comme lors de la résolution du système linéaire, il est préférable de diminuer l'influence des pixels saturés. L'équation précédente devient alors :

$$\ln E_i = \frac{\sum_{j=1}^{P} w(Z_{i,j})(g(Z_{i,j}) - \ln \Delta t_j)}{\sum_{j=1}^{P} w(Z_{i,j})}$$

Lors de l'utilisation de cette formule, il faut être vigilant par rapport aux p Pixels i qui seraient saturées sur toutes les images originales. La somme $\sum_{j=1}^{P} w(Z_{i,j})$ serait alors nulle. Dans ce cas, il suffit juste de ne considérer qu'une image (sans le facteur de poids w) pour le calcul de l'irradiance. Par ailleurs, en pratique, on peut conserver le logarithme pour stocker l'image

HDR.

3.2. Tone mapping

L'image HDR obtenue n'est pas affichable en l'état (sauf sur un écran HDR).

Il faut donc convertir cette image de float 32 bits (ou double 64 bits) à unsigned char 8 bits. Cette étape s'appelle le tone mapping et il existe une multitude de fonctions permettant de réaliser cette compression. Nous commencerons par la fonction la plus simple : l'interpolation linéaire.

$$[x_{min}, x_{max}] \mapsto [Z_{min}, Z_{max}] : x = L(x)$$

Cette fonction consiste à transformer la plus grande valeur de l'image HDR à 255 et la plus petite à 0. En pratique, on pourra copier le résultat de chaque pixel sur les 3 canaux d'une image RGB ce qui générera une image en niveaux de gris.

Cette image contient effectivement une dynamique plus large que celle des images de départ.

4. Et la couleur ?

Pour commencer, nous travaillerons sur images en niveaux de gris. Ainsi, si les images de départs sont en couleur, nous les convertirons en images en niveaux de gris. Pour générer une image HDR en couleurs, les choses se compliquent un peu. On calcule séparément les 3 images HDR correspondant aux 3 canaux. On obtient alors 3 courbes de réponse définies chacune à un facteur prés, mais satisfaisant la contrainte des pixels de couleur $(Z_{mid}, Z_{mid}, Z_{mid})$ sur les images de départ, qui apparaissent achromates sur l'image HDR d'arrivée. Si la fonction de tone mapping est dépendante du contenu de l'image HDR, ce qui est le cas pour la fonction linéaire définie plus haut, les 3 images HDR seront susceptibles d'avoir un minimum et un maximum différents. Ainsi, les valeurs $(Z_{mid}, Z_{mid}, Z_{mid})$ des images de départ apparaitront en niveau de gris sur l'image d'arrivée, mais cela ne sera nécessairement pas le cas des autres valeurs (u, u, u) avec $u \neq Z_{mid}$.

Optionnel : avoir une fonction permettant de traiter les images RGB.

5. HDR sous Matlab

makehdr

Create high dynamic range image

Syntax

```
HDR = makehdr(files)
HDR = makehdr(files, param1, val1,...)
```

Description

`HDR = makehdr(files)` creates the single-precision, high dynamic range image from the set of spatially registered low dynamic range images listed in the `files` cell array. `makehdr` uses the middle

exposure between the brightest and darkest images as the base exposure for the high dynamic range calculations. (This value does not need to appear in any particular file. For more information about calculating this middle exposure value, see Algorithm.)

Note: When you call `makehdr` with this syntax, the low dynamic range image files must contain `exif` exposure metadata.

`HDR = makehdr(files, param1, val1,...)` creates a high dynamic range image from the low dynamic range images in `files`, specifying parameters and corresponding values that control various aspects of the image creation. Parameter names can be abbreviated and case does not matter.

Note: Only one of the `BaseFile`, `ExposureValues`, and `RelativeExposure` parameters may be used at a time.

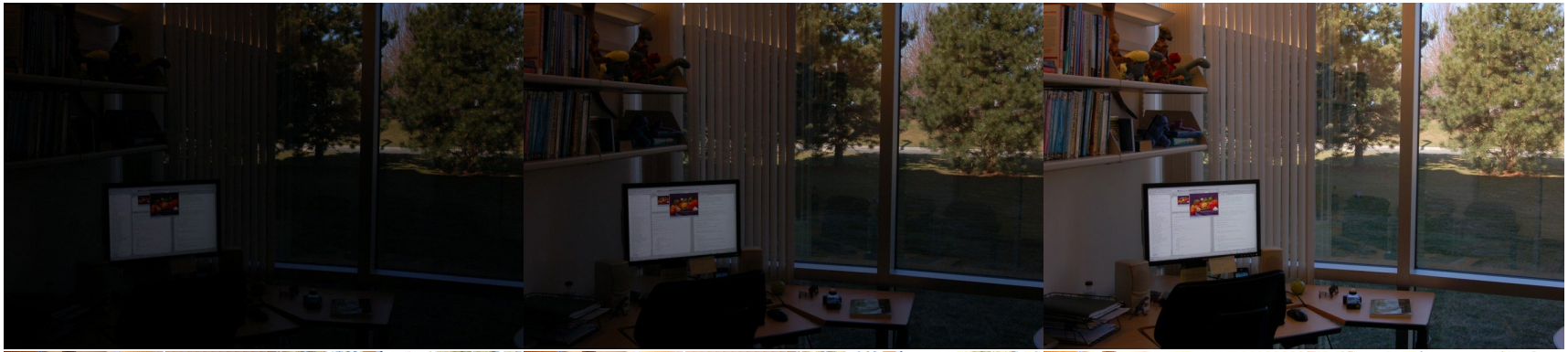
Parameter	Description
`'BaseFile'`	Character array containing the name of the file to use as the base exposure.
`'ExposureValues'`	Vector of exposure values, with one element for each low dynamic range image in the cell array `files`. An increase in one exposure value (EV) corresponds to a doubling of exposure, while a decrease in one EV corresponds to a halving of exposure. Any positive value is allowed. This parameter overrides EXIF exposure metadata.
`'RelativeExposure'`	Vector of relative exposure values, with one element for each low dynamic range image in the cell array `files`. An image with a relative exposure (RE) of 0.5 has half as much exposure as an image with an RE of 1. An RE value of 3 has three times the exposure of an image with an RE of 1. This parameter overrides EXIF exposure metadata.
`'MinimumLimit'`	Numeric scalar value in the range `[0 255]` that specifies the minimum correctly exposed value. For each low dynamic range image, pixels with smaller values are considered underexposed and will not contribute to the final high dynamic range image.
`'MaximumLimit'`	Numeric scalar value in the range `[0 255]` that specifies the maximum correctly exposed value. For each low dynamic range image, pixels with larger values are considered overexposed and will not contribute to the final high dynamic range image.

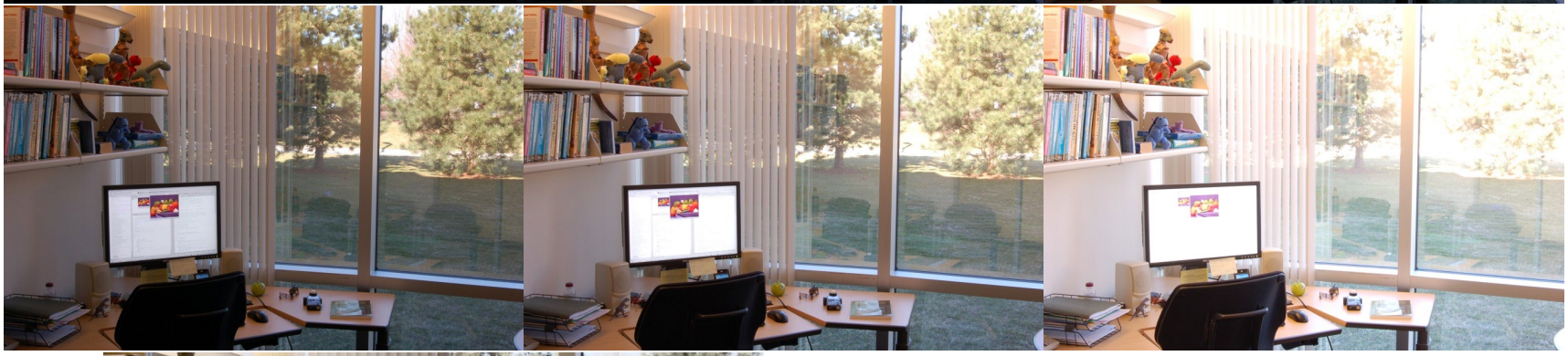
Examples

Make a high dynamic range image from a series of six low dynamic range images that share the same f/stop number and have different exposure times. Use `tonemap` to visualize the HDR image.

```
files = {'office_1.jpg', 'office_2.jpg', 'office_3.jpg', ...
         'office_4.jpg', 'office_5.jpg', 'office_6.jpg'};
expTimes = [0.0333, 0.1000, 0.3333, 0.6250, 1.3000, 4.0000];

hdr = makehdr(files, 'RelativeExposure', expTimes ./ expTimes(1));
rgb = tonemap(hdr);
figure; imshow(rgb)
```





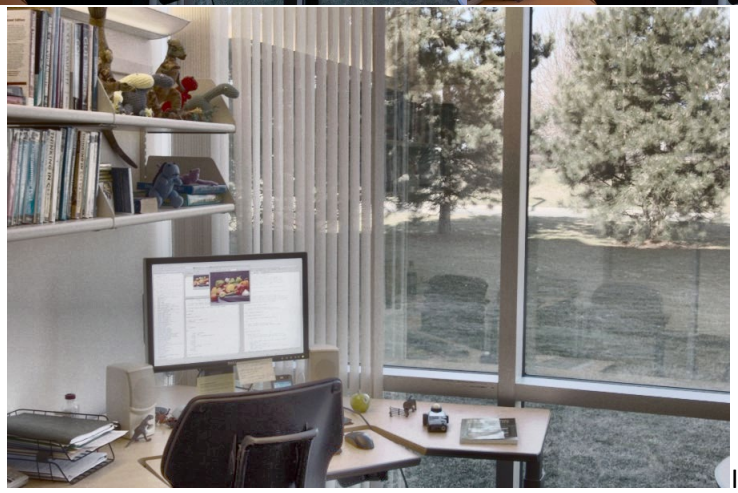


Image HDR obtenue par Matlab à partir des 6 images Ci-dessous