

# High Dynamic Range (HDR)

Topic 3

Week 2 – Jan. 16<sup>th</sup>, 2019

# Topic 3: HDR

- **High Dynamic Range (HDR)**
- Capturing HDR Images
- Computing the Camera Response Function

# Dynamic Range

- Our visual perception is markedly different than that we see in photos
- Highlighted by trying to photograph scenes with very bright & very dark areas
- **Dynamic range**: ratio between the brightest & darkest areas of image visible
- Measured in “**stops**” (like EV):  
 $\log_2(\text{dynamic range})$



Debevec et al, SIGGRAPH'97

# Dynamic Range

- Human: ~14 EV stops
- 8-bit image: ~6 stops
- EV is a log scale!
- We can't capture HDR well
  - Best sensors capture 14 bits/pixel
  - Most images are 8-bit
- We can't display HDR well
  - TV/monitors: 6-10 Stops



Debevec et al, SIGGRAPH'97

(T. Borer et al. ["A \"Display Independent\" High Dynamic Range Television System\"](#))

# Capturing HDR: The Problem

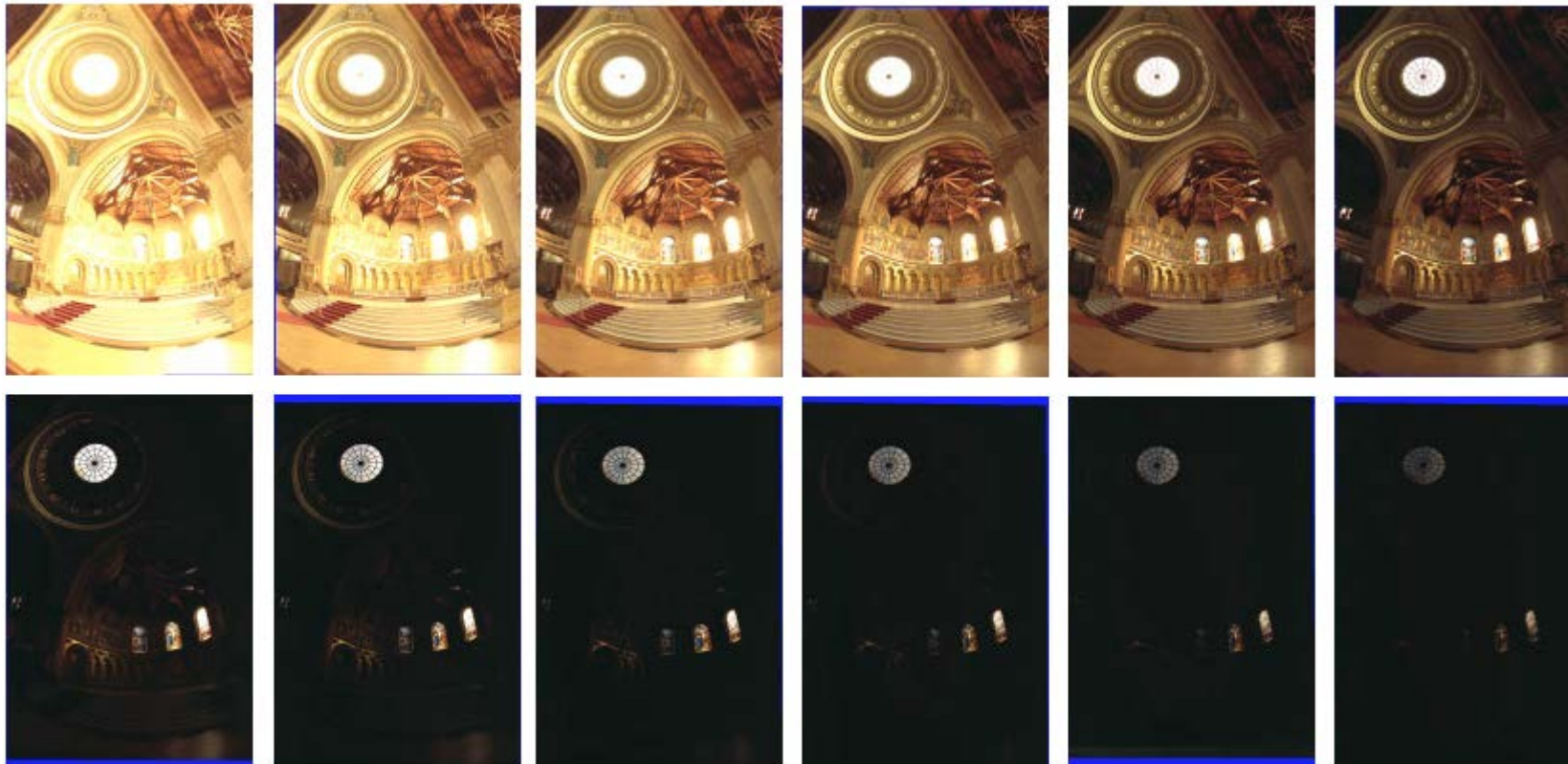
- Bright pixels: saturate the sensor – no information!
- Dark pixels: are below the threshold required to be represented





# Capturing HDR: Multiple Exposures

- Idea: Take photos at multiple EV to capture more dynamic range
  - For low EV, bright areas won't saturate pixels
  - For high EV, dark areas will have give enough light to be representable



# Displaying HDR: Tone Mapping

- Idea: Like we use a gamma function to better display linear images, find a mapping function that can display more tones from our HDR image in 8-bit LDR display
- Making this look realistic is perhaps the bigger challenge!



# HDR in 2019

- A lot has changed since Fattal et al. in 2002
- Almost every modern smartphone has an HDR capture mode (of various extent/quality)
- High-end TVs/monitors now sell with “HDR” feature, movies are increasingly captured in HDR (both of these 10-bit, ~10 stops!)



# HDR in 2019

- Almost every new smartphone has an HDR capture mode (of various extent/quality)

## Burst photography for high dynamic range and low-light imaging on mobile cameras

Samuel W. Hasinoff  
Jonathan T. Barron

Dillon Sharlet  
Florian Kainz  
Google Research

Ryan Geiss  
Jiawen Chen

Andrew Adams  
Marc Levoy



**Figure 1:** A comparison of a conventional camera pipeline (left, middle) and our burst photography pipeline (right) running on the same cell-phone camera. In this low-light setting (about 0.7 lux), the conventional camera pipeline underexposes (left). Brightening the image (middle) reveals heavy spatial denoising, which results in loss of detail and an unpleasantly blotchy appearance. Fusing a burst of images increases the signal-to-noise ratio, making aggressive spatial denoising unnecessary. We encourage the reader to zoom in. While our pipeline excels in low-light and high-dynamic-range scenes (for an example of the latter see figure 10), it is computationally efficient and reliably artifact-free, so it can be deployed on a mobile camera and used as a substitute for the conventional pipeline in almost all circumstances. For readability the figure has been made uniformly brighter than the original photographs.

# HDR in 2019

- High-end TVs now sell with “HDR” of varying quality
- Movies are captured in HDR
- Both of these are at best 10-bit, ~10 stops!
- Still far from human DR!

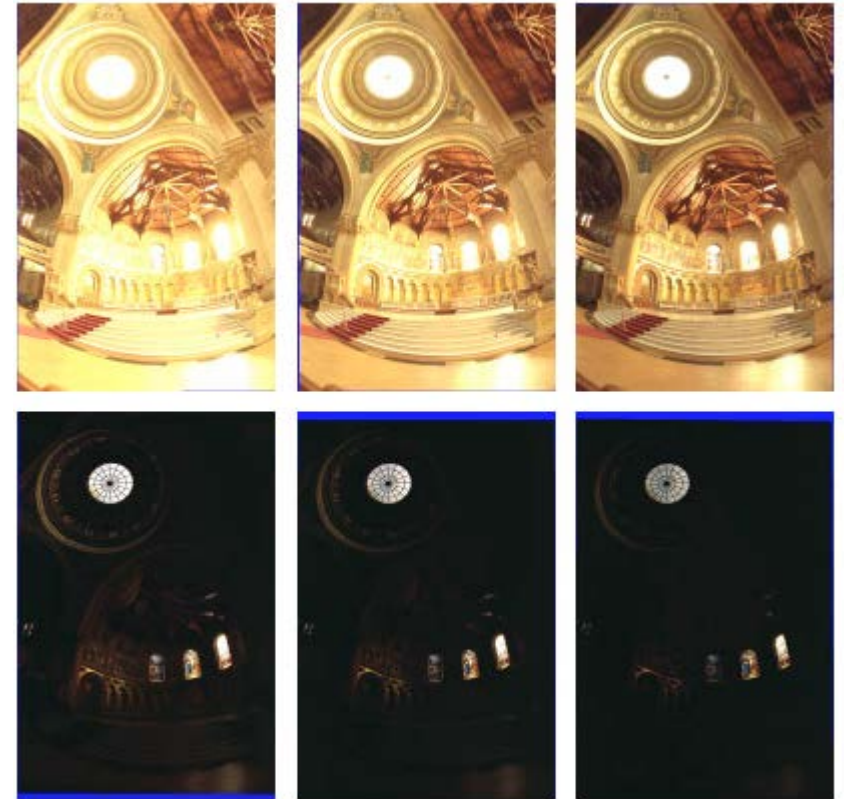


# Topic 3: HDR

- High Dynamic Range (HDR)
- **Capturing HDR Images**
- Computing the Camera Response Function

# Capturing HDR: Idea

- Problem: Our cameras can only capture a limited representation of the scene irradiance  $\Phi$ , represented by pixel value  $Z$
- In different exposures, a different dynamic range for each pixel is captured
- The true scene irradiance for each pixel will only be represented within the dynamic range of some of these photos
- Idea: Let's capture many exposures and somehow combine them!





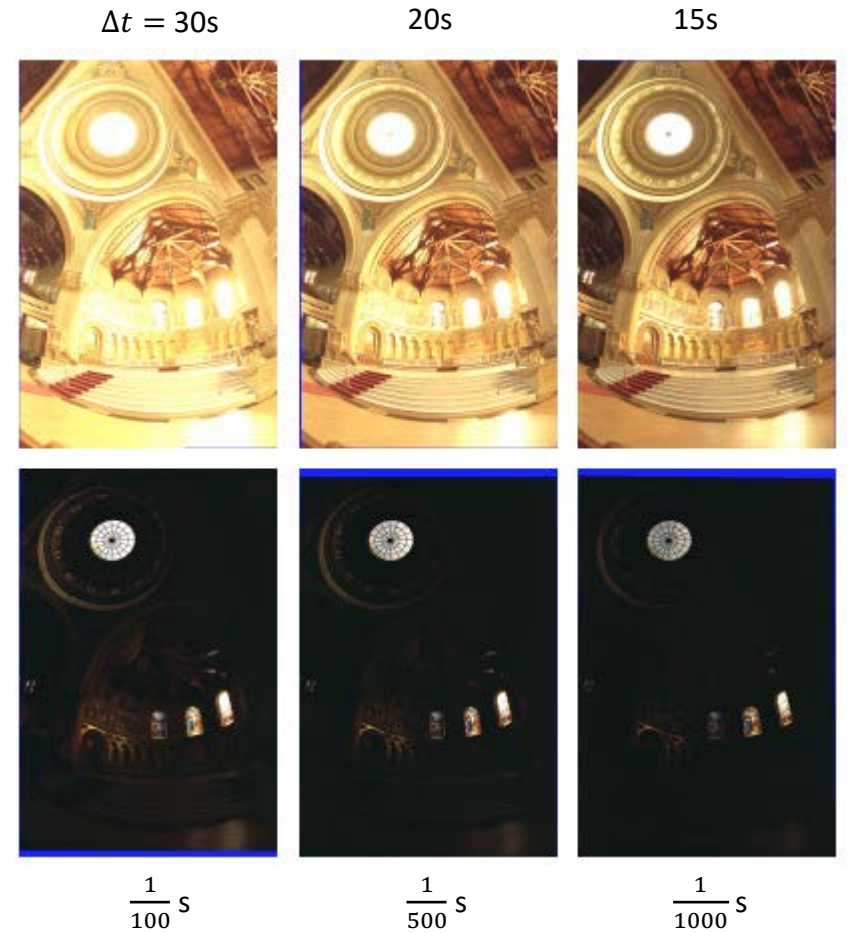
# Capturing HDR: Multiple Exposures



- What camera settings should we use capture the different exposures?

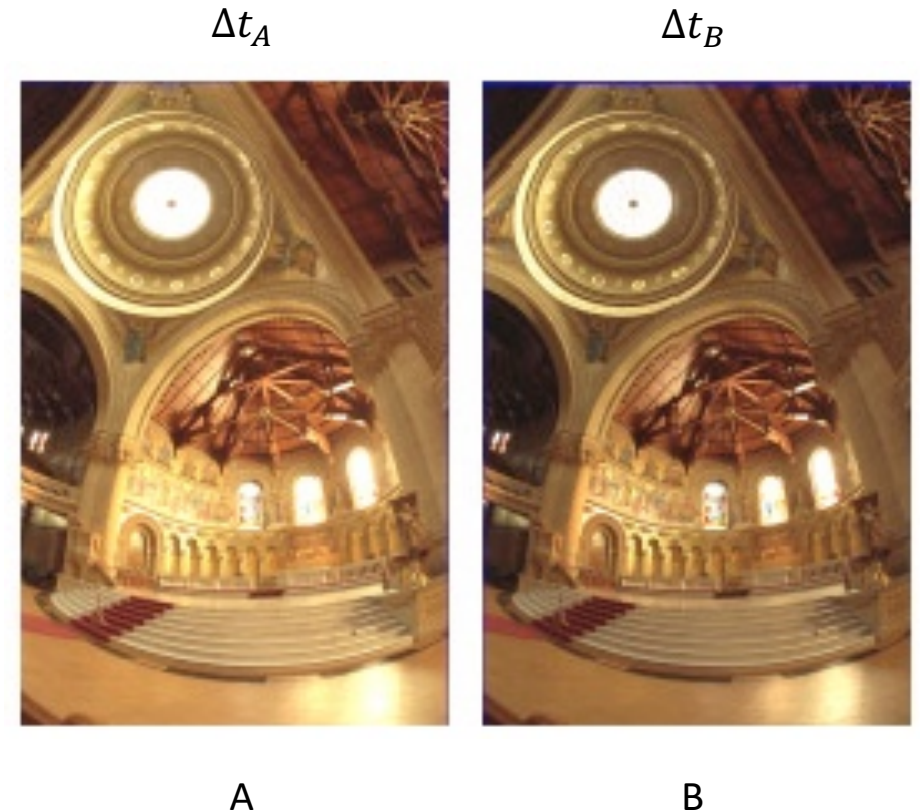
# Capturing HDR: Multiple Exposures

- Capture with different exposure times  $\Delta t$
- Assume view doesn't change (i.e. camera is on tripod, or steady), and scene is static
- To combine, we need to know how pixel value in each image relates!



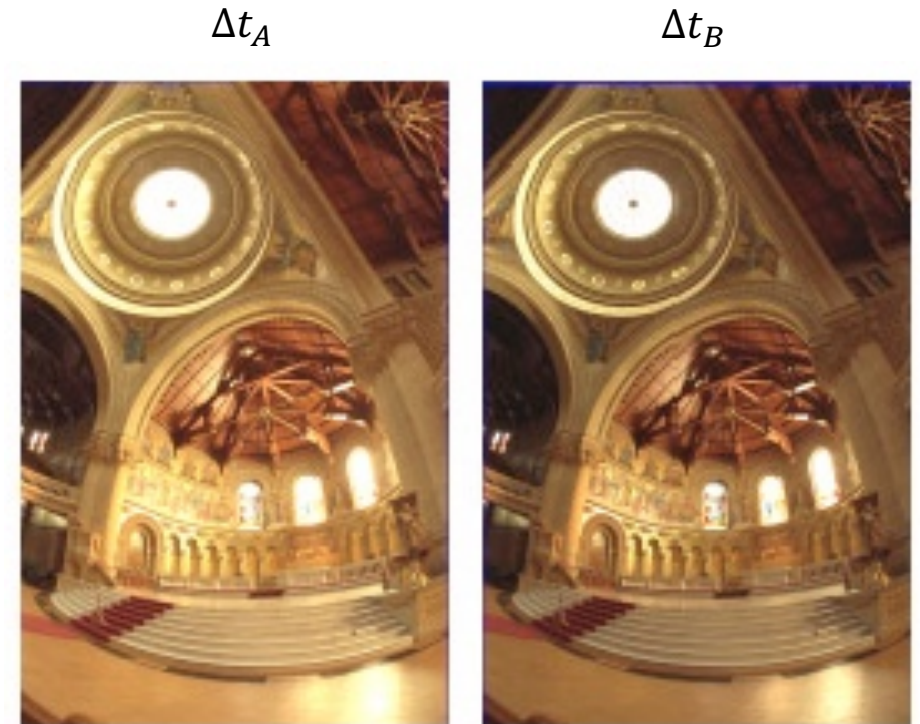
# Capturing HDR: Multiple Exposures

- Assume we have two photos, A and B, where  $\Delta t_A = 2\Delta t_B$
- i.e. B has half the exposure duration
- How does each pixel  $Z_B(x, y)$  in B relate to corresponding pixel  $Z_A(x, y)$  in A?



# Capturing HDR: Multiple Exposures

- $Z_A(x, y), Z_B(x, y)$ , where  $\Delta t_A = 2\Delta t_B$
- If we assume the camera response function is **linear** then,
  - $Z_A(x, y) = 2 Z_B(x, y)$
- **However**, we pass the pixel's response through a non-linear gamma function when converting from linear RAW image! (see Topic 1)
  - If we have the RAW photos we are set
  - What if we don't?

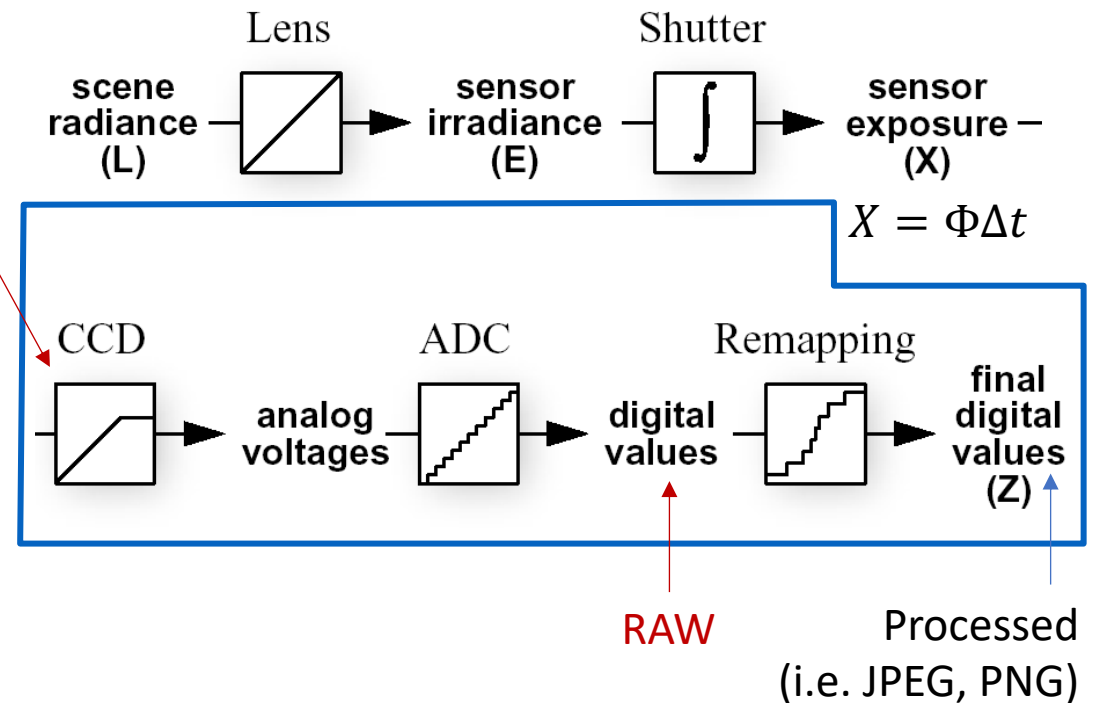




# Inverse Camera Response Function

- With RAW, response is linear (up to  $I_m$ )
- Without RAWs, knowing the relative exposure of two photos relies on us knowing the camera response function:  $f_{\text{camera}}$
- Recall:  $f_{\text{camera}}$  gives the pixel value ( $Z$ ) for a given scene irradiance:

$$Z = f_{\text{camera}}(\Phi\Delta t)$$

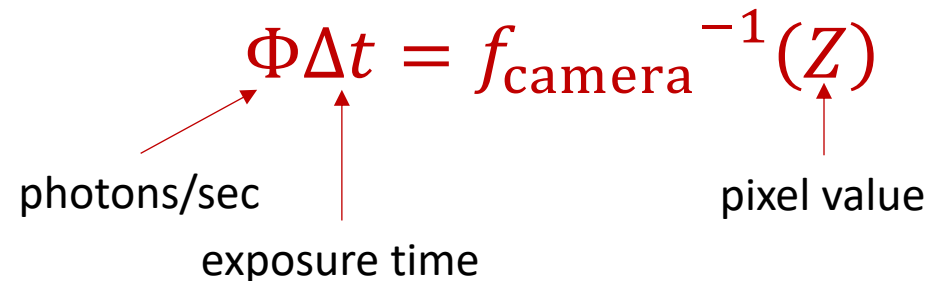


# Inverse Camera Response Function

- Recall C.R.F. gives the pixel value ( $Z$ ) for a given scene irradiance:

$$Z = f_{\text{camera}}(\Phi\Delta t)$$

- We want the scene irradiance given the pixel value ( $Z$ ), i.e. the inverse C.R.F.:

$$\Phi\Delta t = f_{\text{camera}}^{-1}(Z)$$


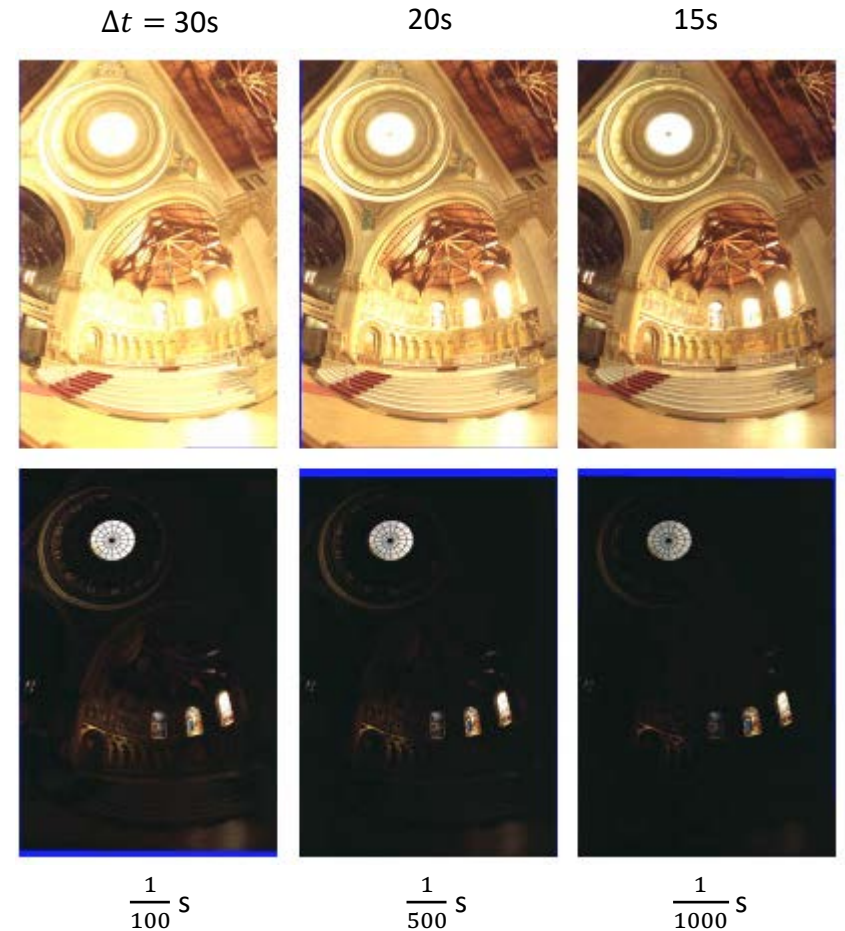
The diagram shows the equation  $\Phi\Delta t = f_{\text{camera}}^{-1}(Z)$  in red. Below the equation, there are three annotations with red arrows pointing to specific parts of the equation: 'photons/sec' points to  $\Phi$ , 'exposure time' points to  $\Delta t$ , and 'pixel value' points to  $Z$ .

# Irradiance from Pixel Value

- Assume we have  $N$  different exposures
- Each exposure  $\Delta t$  gives us an estimate of the irradiance  $\Phi$  for the pixel value  $Z$ :

$$\Phi \Delta t = f_{\text{camera}}^{-1}(Z)$$

- Some of these estimates will be better than others!
  - i.e. estimates from very dark/bright pixels will be poor (why?)



# Merging Multiple Exposures: Debevec et al.

- Algorithm Outline: (see Debevec et al. in readings)

for each pixel location  $i$  and pixel value  $Z_i$ :

for each photo  $j = 1, \dots, P$  with exposure time  $\Delta t_j$ :

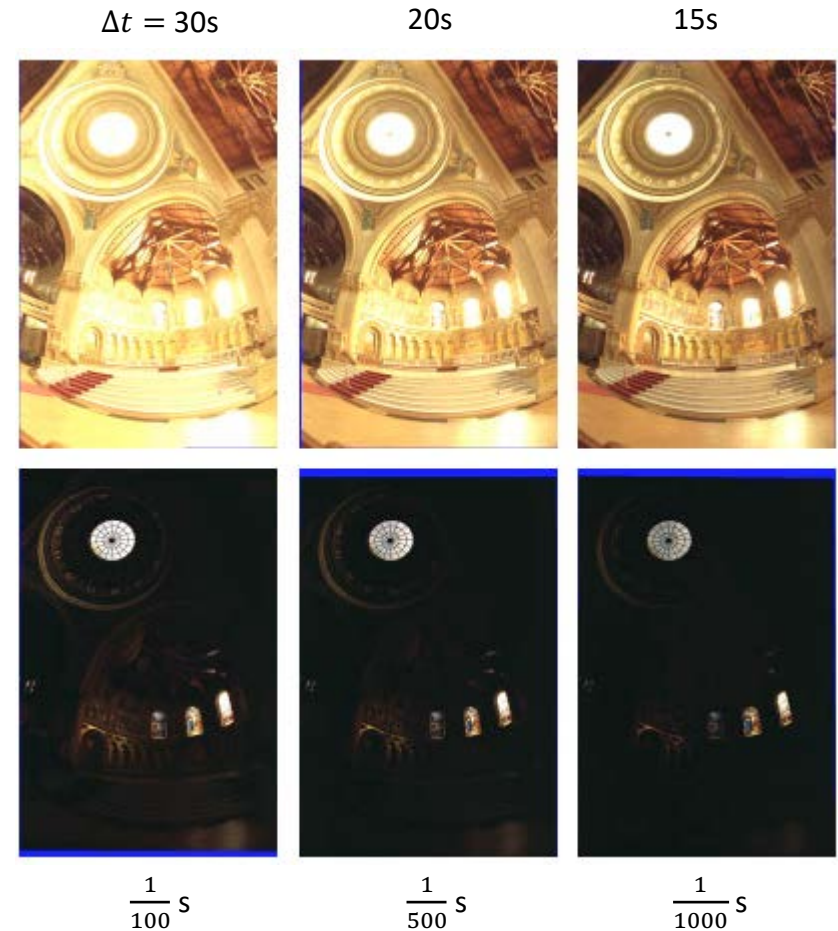
estimate  $\Phi_{ij}$  at pixel location  $i$  given  $\Delta t_j, Z_i$

given that inverse camera response function is known

combine estimates  $\Phi_{ij}$ , where  $j = 1, \dots, P$  to get  $\Phi_i$

Result: HDR image where each pixel is a float based on our estimate of the scene irradiance  $\Phi_i$  for each pixel

- $i$  is the pixel location, regarding image as flattened 1D array of pixels



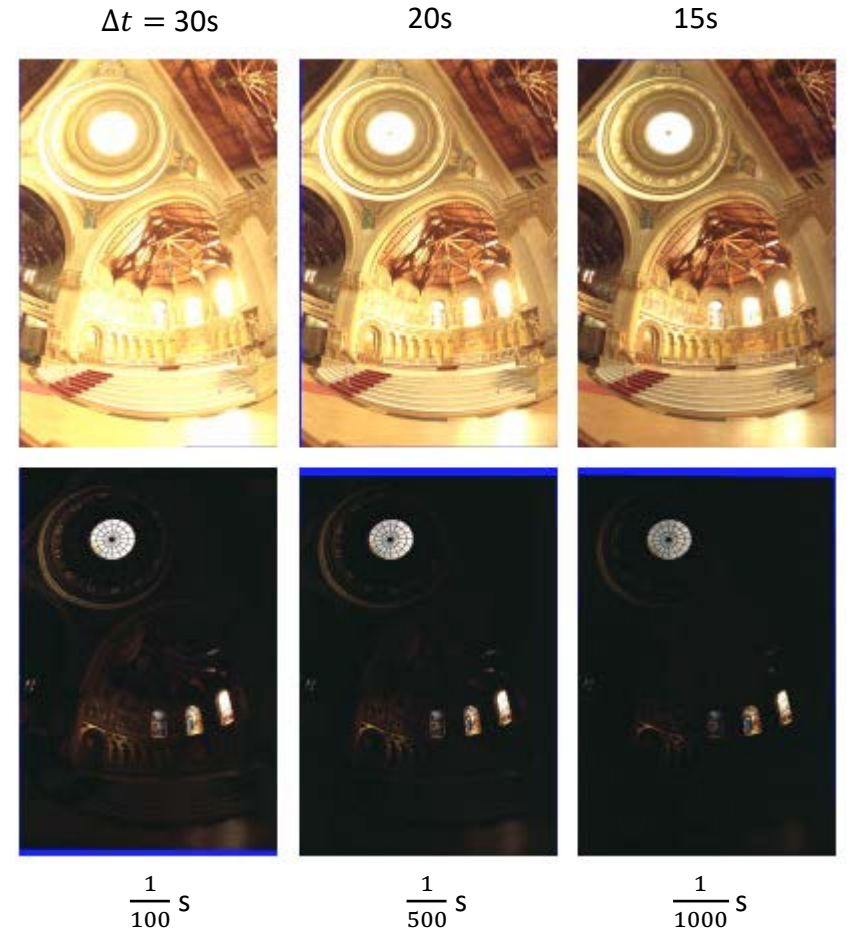


# Topic 3: HDR

- High Dynamic Range (HDR)
- Capturing HDR Images
- **Computing the Camera Response Function**

# Estimating the Camera Response Function

- Want to estimate the function:  
$$\Phi \Delta t = f_{\text{camera}}^{-1}(Z)$$
- For every photo we take, we get samples of  $Z$  and  $\Delta t$
- **Problem:** we have no way of measuring  $\Phi$ !
- **Solution:** we have a lot of data, and a couple of tricks!

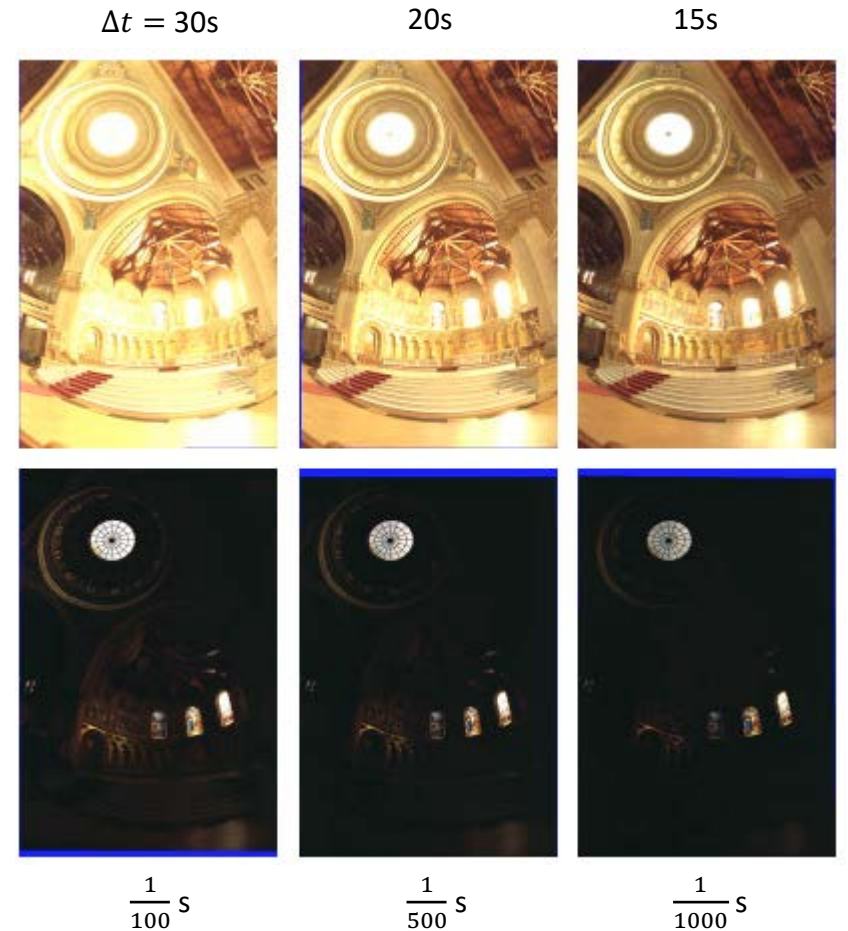


# Estimating the Camera Response Function

$$\Phi \Delta t = f_{\text{camera}}^{-1}(Z)$$

How do these quantities change by pixel/photo?

- $f_{\text{camera}}$ : same for all pixels & photos
- $Z$ : different for each pixel and image
- $\Delta t$ : different for each photo, same for pixels in the same photo
- $\Phi$ : same for corresponding pixel location across different photos (same scene irradiance), different across pixel locations



# Trick #1: Log-Inverse Response Function

- We know how to solve linear systems of equations, however,

$$f^{-1}(Z) = \Phi \cdot \Delta t$$

isn't linear, i.e.:

$$a_1 x_1 + a_2 x_2 + \dots + a_N x_N = b$$

- Instead we can represent it using log as:

$$\log f^{-1}(Z) = \log \Phi + \log \Delta t$$

- We will denote  $g(Z) = \log f^{-1}(Z)$  as the **log-inverse response function**:

$$g(Z) = \log \Phi + \log \Delta t$$



# Trick #2: Discrete Function Approximation

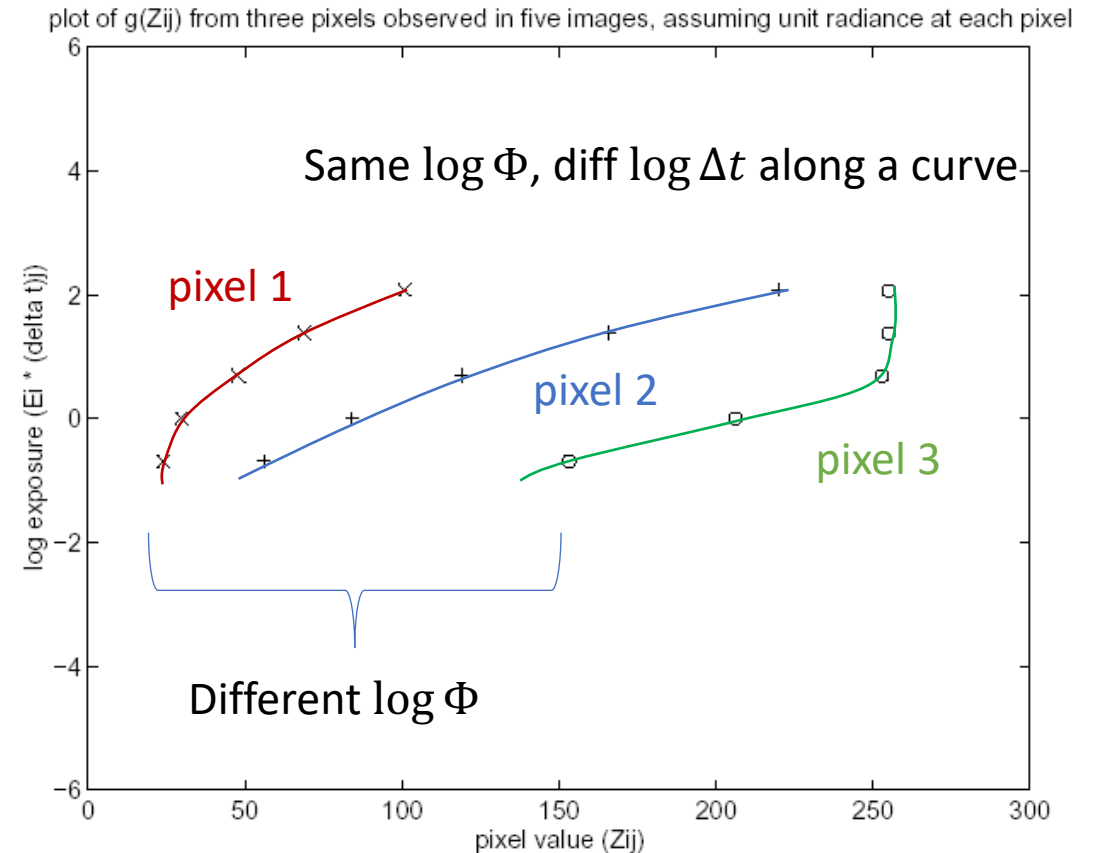
- Our pixels only have 256 **discrete** values, i.e.  $Z \in \mathbb{Z}: 0 \leq Z \leq 255$
- Thus we can treat each value of  $Z$  as a different equation, i.e.:

$$\begin{aligned} g(Z = 0) &= \log \Phi + \log \Delta t \\ g(Z = 1) &= \log \Phi + \log \Delta t \\ &\vdots \\ g(Z = 255) &= \log \Phi + \log \Delta t \end{aligned}$$

# Computing the Log-Inverse Response Fcn.

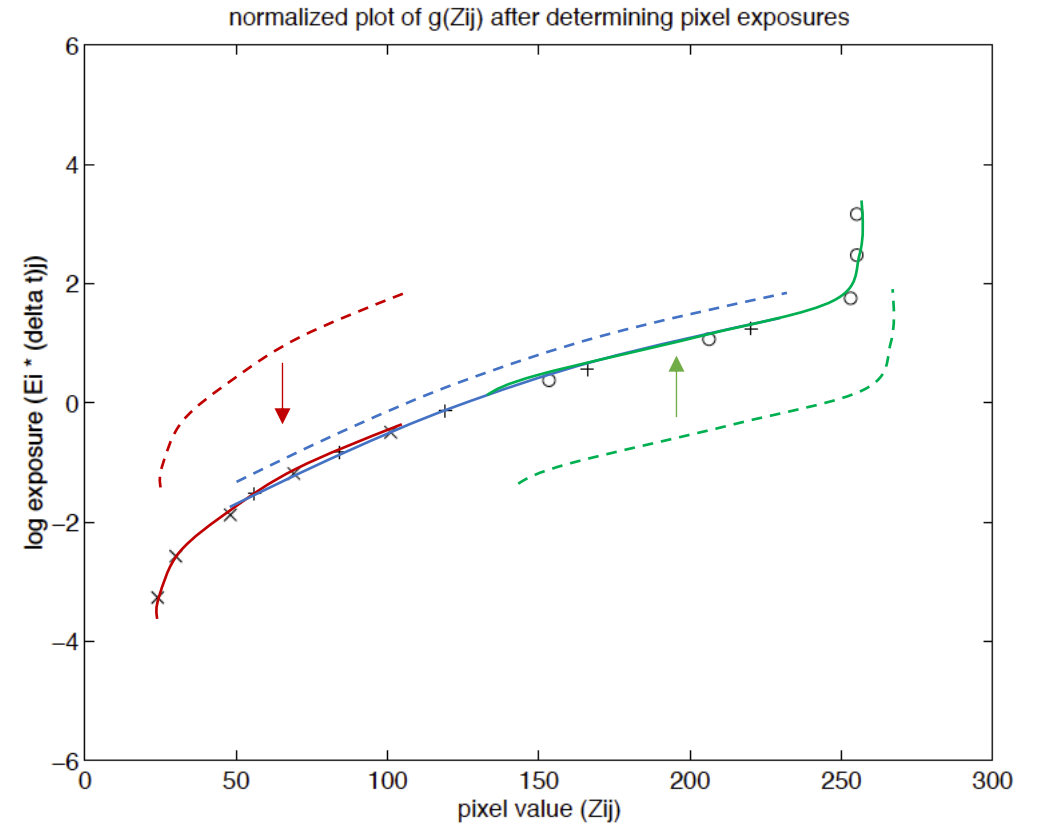
- Plot samples of pixel values  $Z$  v.s.  $\log \Delta t + \log \Phi$  for 3 different pixel locations
- However, we know different pixel locations have a different  $\log \Phi$ !
- How will the samples change with different values of  $\log \Phi$ ?

$$g(Z) = \log \Phi + \log \Delta t$$



# Computing the Log-Inverse Response Fcn.

- With the correct relative  $\log \Phi$  for each pixel location, the samples line up



# Putting it all together

For each pixel location  $1 \leq i \leq N$ , photo  $1 \leq j \leq P$ :

Diagram illustrating the equation  $g(Z_{ij}) = \log \Phi_i + \log \Delta t_j$  with annotations:

- pixel value for  $i^{\text{th}}$  pixel,  $j^{\text{th}}$  image (**known**) points to  $Z_{ij}$
- pixel scene irradiance for  $i^{\text{th}}$  pixel (**unknown**) points to  $\Phi_i$
- inverse log C.R.F. (**unknown**) points to  $g$
- exposure time of  $j^{\text{th}}$  image (**known**) points to  $\Delta t_j$

We want to compute:

- $g(0), g(1), \dots, g(255)$  255 unknowns
- $\log \Phi_i$  N unknowns

We are given:

N\*P equations

- N pixel locations + intensity in P images, with known  $\Delta t_j$

# Putting it all together: Equations + Unknowns

For each pixel location  $1 \leq i \leq N$ , photo  $1 \leq j \leq P$ :

Diagram illustrating the equation  $g(Z_{ij}) = \log \Phi_i + \log \Delta t_j$  with annotations:

- pixel value for  $i^{\text{th}}$  pixel,  $j^{\text{th}}$  image (**known**) points to  $Z_{ij}$
- pixel scene irradiance for  $i^{\text{th}}$  pixel (**unknown**) points to  $\Phi_i$
- inverse log C.R.F. (**unknown**) points to  $g$
- exposure time of  $j^{\text{th}}$  image (**known**) points to  $\Delta t_j$

We are given:

- **$N$  pixels in  $P$  images**, with known  $\Delta t_j$

We know:

$$\forall_{ij}, g(Z_{ij}) - \log \Phi_i = \log \Delta t_j$$

Idea:

- Each pixel in each photo is one equation, so we have  **$N \cdot P$  equations!**
- Unknowns:  $\Phi_i, g \rightarrow N + 256$  **unknowns**



# Equations

- Each pixel:  $g(Z_{ij}) - \log \Phi_i = \log \Delta t_j$
- Let (to simplify notation):
  - $g_{Z_{ij}} \equiv g(Z_{ij})$ ,
  - $e_i \equiv \log \Phi_i$
  - $\delta_j \equiv \log \Delta t_j$
- Then, for each pixel:
  - $g_{ij} - e_i = \delta_j$
- Single eqn. in matrix form:

$$\begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 & -1 & \dots & 0 \end{bmatrix} \begin{bmatrix} g_0 \\ \vdots \\ g_{255} \\ e_1 \\ \vdots \\ e_N \end{bmatrix} = \delta_j$$

$(Z_{ij}+1)\text{-th column} \quad \uparrow \quad (256+i)\text{-th column}$

# Equations

- System of equations in matrix form:

If pixel  $i$   
has intensity  
 $Z_{ij}$  in image  $j$

$(ij)$ -th row  $\rightarrow$

$$\begin{array}{c}
 \xleftarrow{256 + N \text{ columns}} \\
 \begin{array}{c}
 \xleftarrow{256 \text{ cols}} \quad \xleftarrow{N \text{ cols}} \\
 \begin{array}{c}
 \text{column } Z_{ij} + 1 \\
 \text{column } 256 + i
 \end{array} \\
 \begin{array}{cccccc}
 0 & 1 & 0 & 0 & -1 & 0
 \end{array} \\
 \begin{array}{c}
 \text{coeffs of} \\
 \text{unknowns } g
 \end{array}
 \quad
 \begin{array}{c}
 \text{coeffs of} \\
 \text{unknowns } e
 \end{array}
 \end{array}
 \end{array}
 \begin{bmatrix}
 g_0 \\
 \vdots \\
 g_{255} \\
 e_1 \\
 \vdots \\
 e_N
 \end{bmatrix}
 =
 \begin{bmatrix}
 \vdots \\
 \delta_j \\
 \vdots
 \end{bmatrix}$$

# Solving The System of Equations

- To solve this system, matrix  $A$  must be non-singular
- We can help ensure this by choosing  $N, P$  such that  $NP \gg 256 + N$
- e.g.  $N=1000$  pixels from  $P = 20$  photos (20 exposure settings)

$$A x = b$$

Diagram illustrating the system of equations  $Ax = b$ . Matrix  $A$  is labeled with dimensions  $N \times P$  rows and  $256 + N$  columns (pixels). The vector  $x$  is labeled with dimensions  $P$  exposure settings. The vector  $b$  is labeled with dimensions  $N$  exposure settings.

# Smoothness Constrains

Idea: The camera response function smoothly in real cameras

So we add more equations to enforce smoothness

Intuition: Force near-constant rate of change:

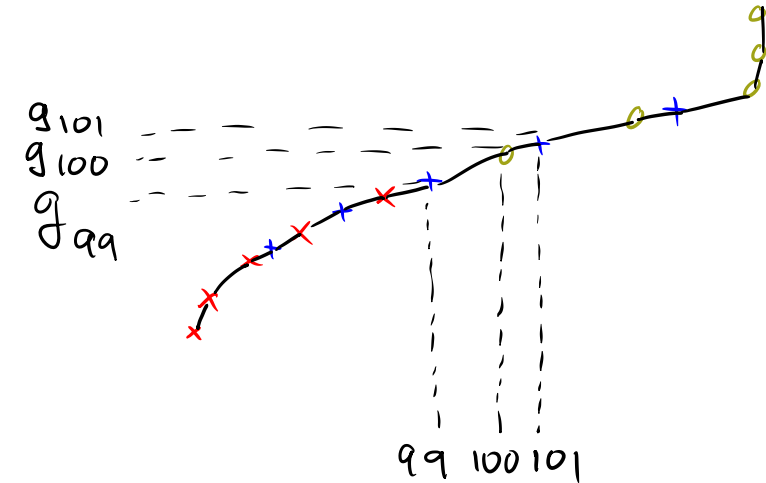
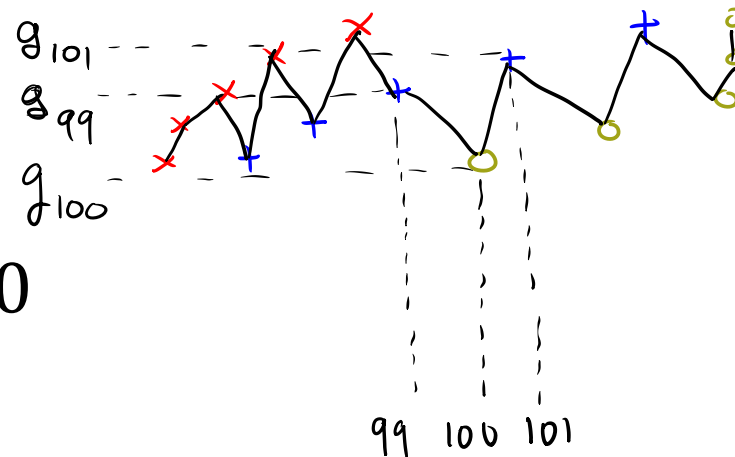
2nd order derivative  $\approx 0$

$$g_{100} - g_{99} \approx g_{101} - g_{100} \Leftrightarrow 2g_{100} - g_{101} - g_{99} \approx 0$$

Add these equations  
to the system:

$$2g_l - g_{l+1} - g_{l-1} = 0$$

$$(l = 1, 2, \dots, 254)$$





# Smoothness Constraints

- Original equations

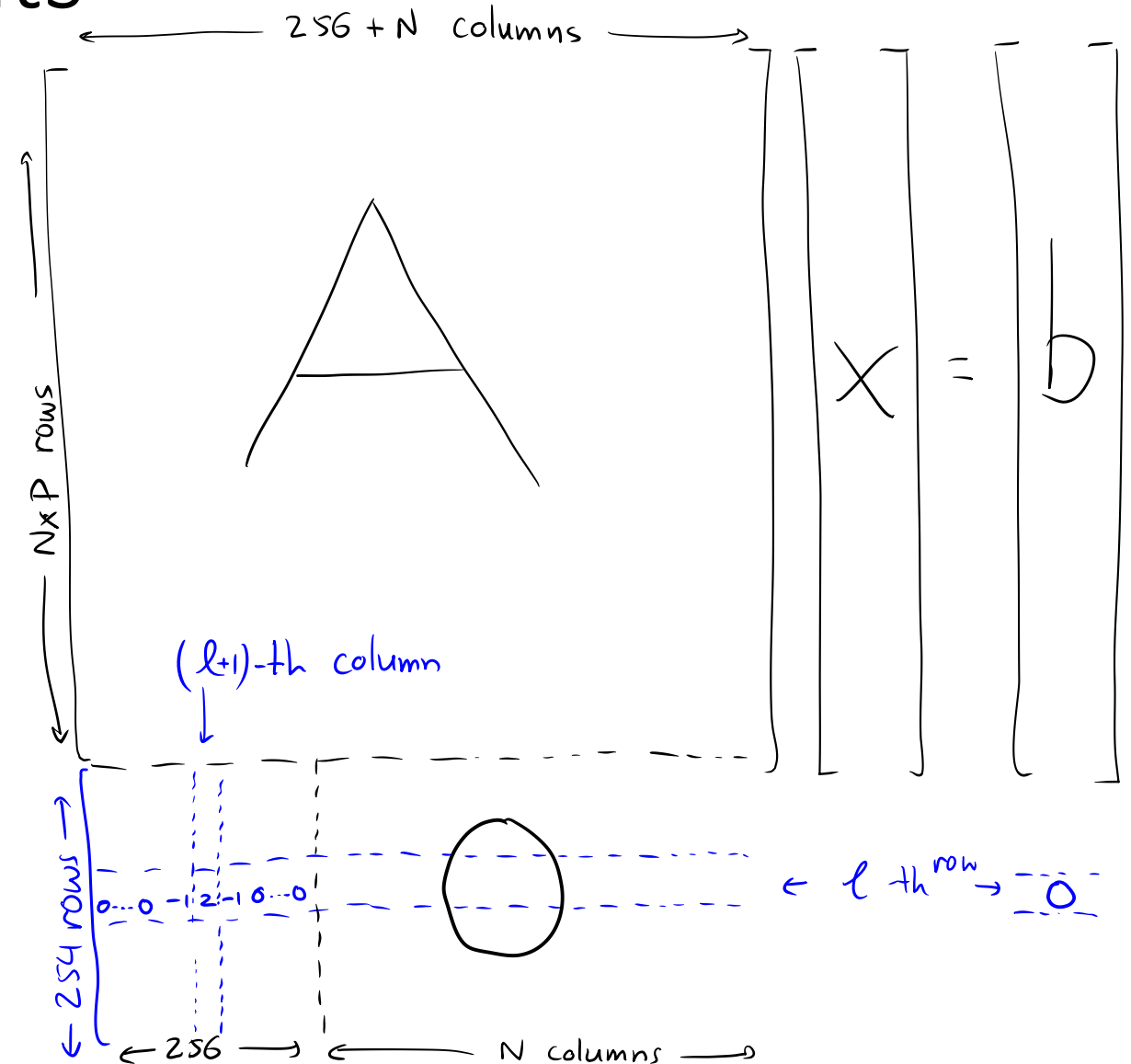
$$g_{ij} - e_i = \delta_j$$

$$(1 \leq i \leq N, 1 \leq j \leq P)$$

- Smoothness equations

$$2g_l - g_{l+1} - g_{l-1} = 0$$

$$(l = 1, 2, \dots, 254)$$



# Finally: Merging Multiple Exposures

- Algorithm HDR: (see Debevec et al. in readings)  
for each pixel location  $i$  and pixel value  $Z_i$ :

for each photo  $j = 1, \dots, P$ :

$$\log \Phi_{ij} = g(Z_{ij}) - \log \Delta t_j$$

$$\log \Phi_i = \frac{\sum_{j=1}^P w(Z_{ij}) \log \Phi_{ij}}{\sum_{j=1}^P w(Z_{ij})} \quad \text{weighted sum ...}$$

Where  $w(Z_{ij})$  is a **weighting factor** that depends on the pixel value. Pixels close to saturation (255), or close to the black level (0) are weighted lower.

(Note: in the paper  $E \equiv \Phi$ )

