HDR images acquisition

dr. Francesco Banterle

francesco.banterle@isti.cnr.it

Current sensors

- No sensors available to consumer for capturing HDR content in a single shot
- Some native HDR sensors exist, HDRc by Omron, but some issues:
  - too much noise
  - low resolution (around 1024x768)
  - expensive to manufacture

Exposure bracketing

- Capturing many LDR images (8-bit) of the same scene:
  - from the darkest area in the scene
  - to the brightest area in the scene
- The scene has to be static!!!

Exposure bracketing

t = 1/128s

t = 1/32s

t = 1/8s

Exposure bracketing

- Required equipment:
  - camera with the possibility to vary the exposure
  - tripod (avoid camera shake)
- Optional equipment:
  - luminance meter
  - colorchecker chart
  - remote control for the camera

How many exposures?

- Brute force approach:
  - Select an exposure for the darkest/brightest area in the scene and take a shot
  - Double/half exposure and take a shot
  - Repeat until brightest/darkest are in the scene is captured

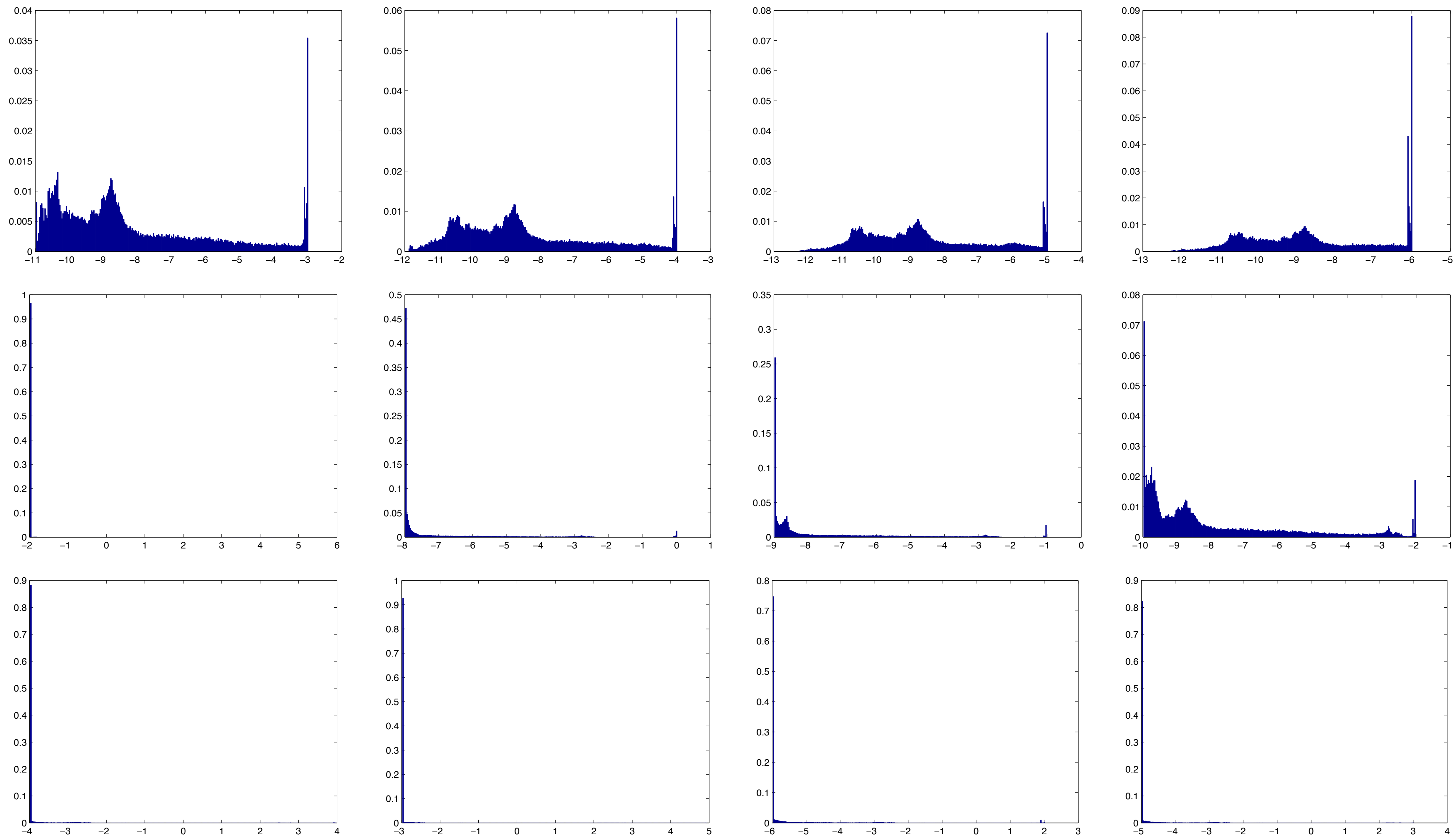
How many exposures?

- Some issue with this approach:
  - time consuming, especially if the camera is not programmable
  - we are making micro movements at every click!
  - over-sampling, maybe there is no need

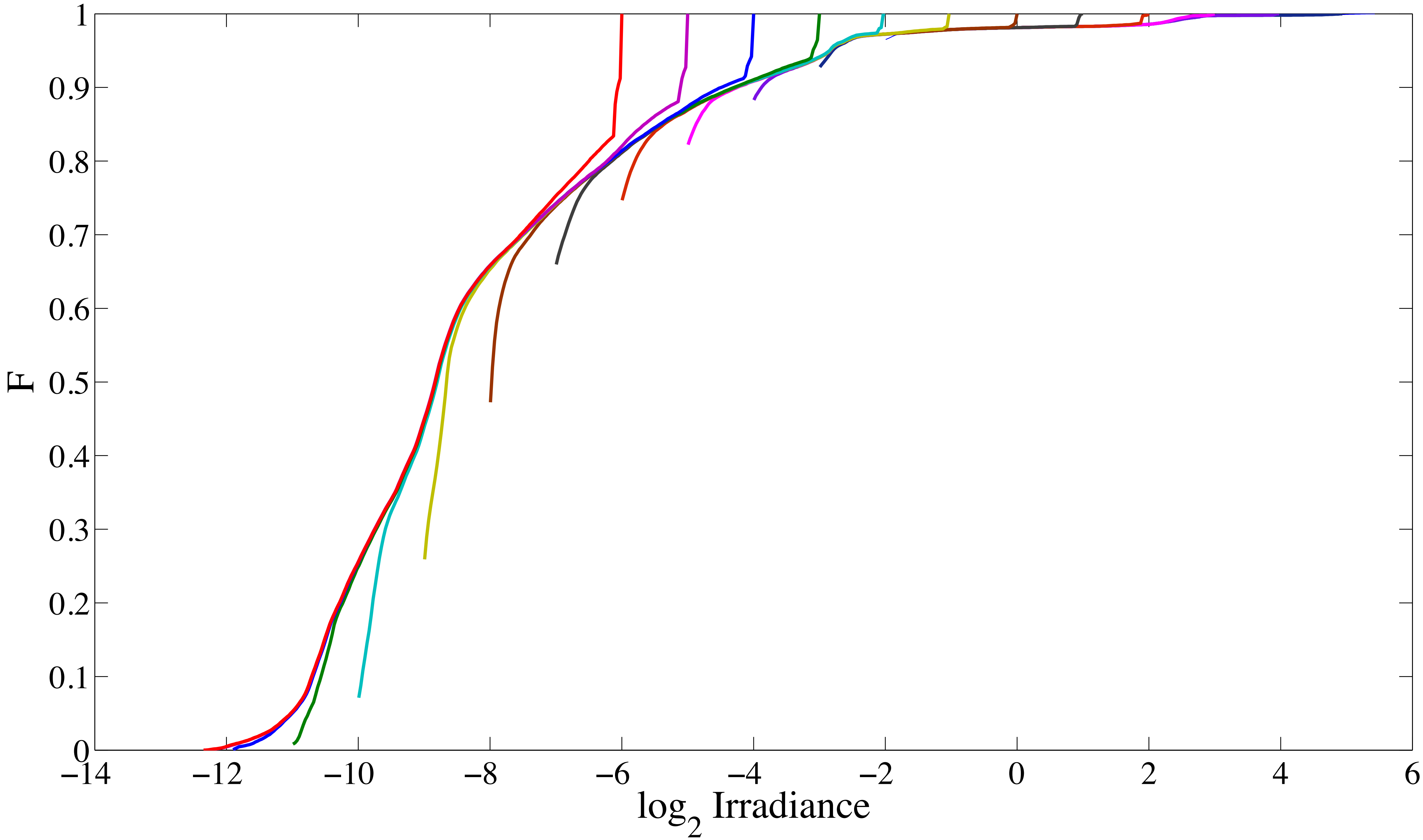
Exposure Metering

- Exposure metering [Gallo et al. 2012]:
  - capturing histograms from the viewfinder (picture preview in a camera) - free!
  - computing CDF for each histogram $F(n) = \frac{\sum_{i=0}^{n} H(i)}{\sum_{i=0}^{N} H(i)}$
  - obtaining the global CDF
  - differentiation —> HDR histogram

Exposure Metering: LDR Histograms



Exposure Metering: LDR CDFs



Exposure Metering

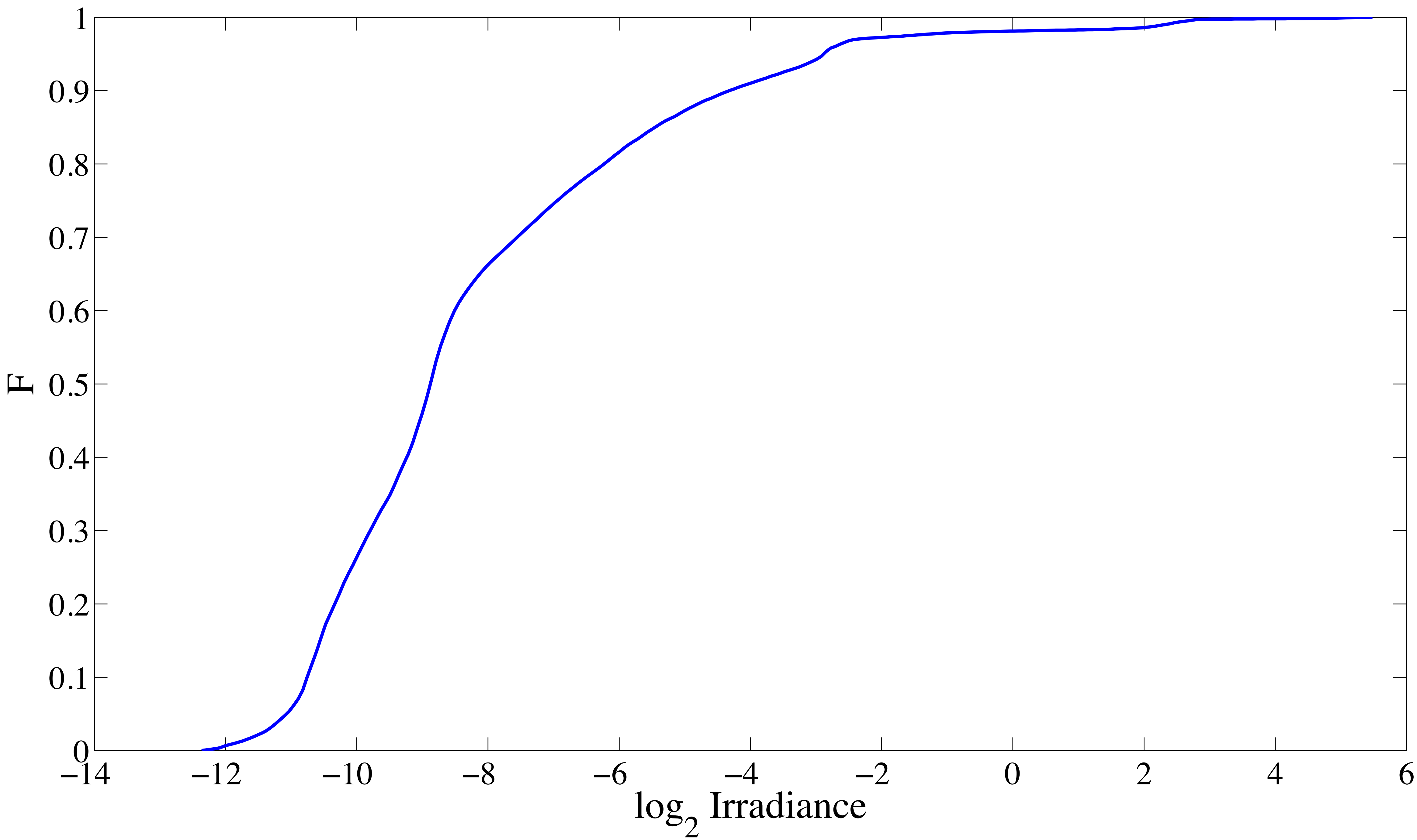
Algorithm 2.1: $\textsc{CompFullCDF}(\{B_k\}, \{b^{\tilde{E}}_{i,j}\}, F^L_1, F^L_2, \ldots, F^L_J)$

for $k \leftarrow 0$ **to** $K - 1$
  do $F^H(B_k) \leftarrow 0$

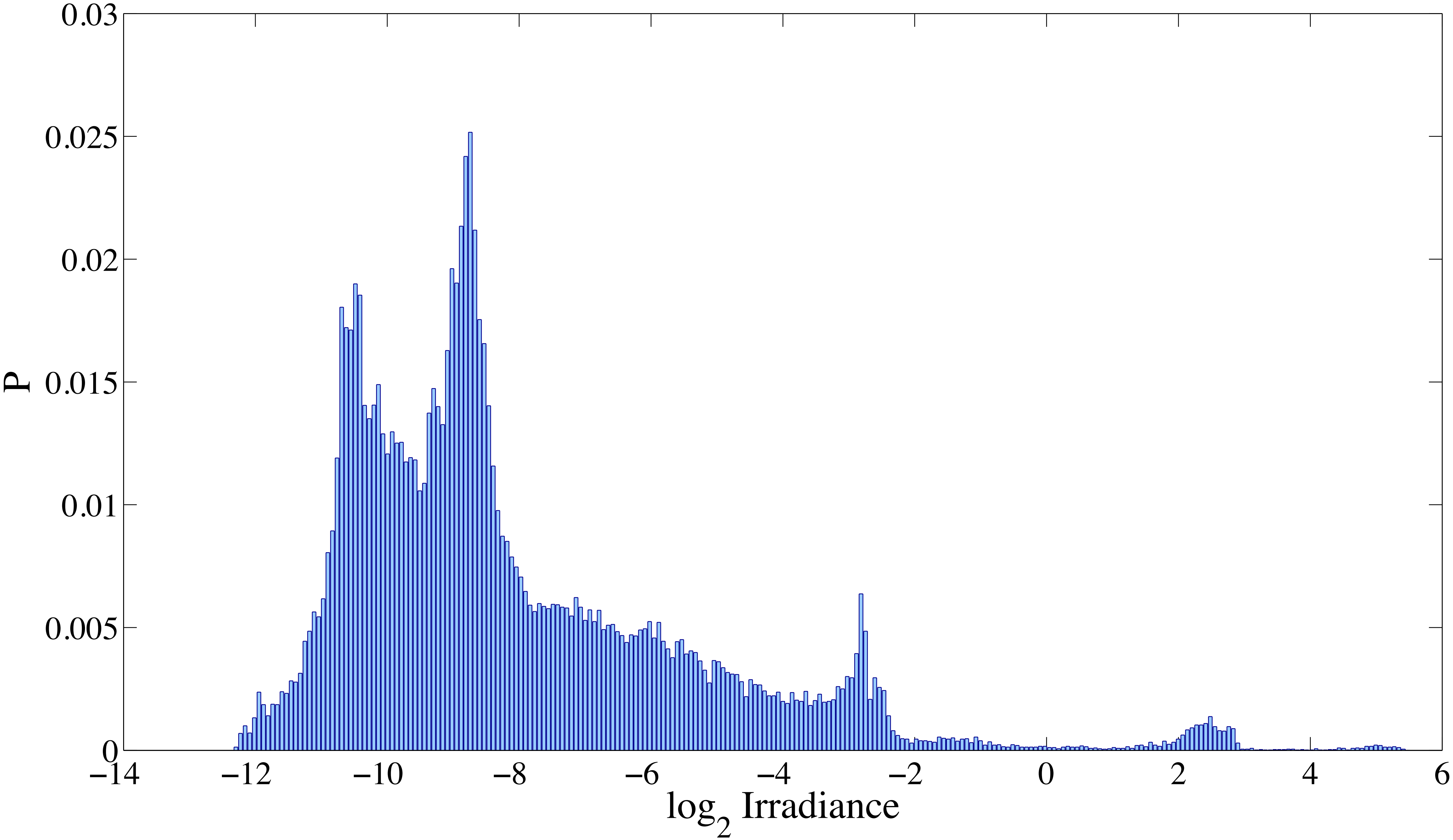
for $j \leftarrow 0$ **to** $J - 1$
  do {
    for $i \leftarrow 0$ **to** $I - 2$
      do {
        for each $k : B_k \in (b^{\tilde{E}}_{i,j}, b^{\tilde{E}}_{i+1,j}]$
          do $F^H(B_k) \leftarrow max(F^H(B_k), F^L_j(b^{\tilde{E}}_{i,j}))$

return (F^H)

Exposure Metering: HDR CDF



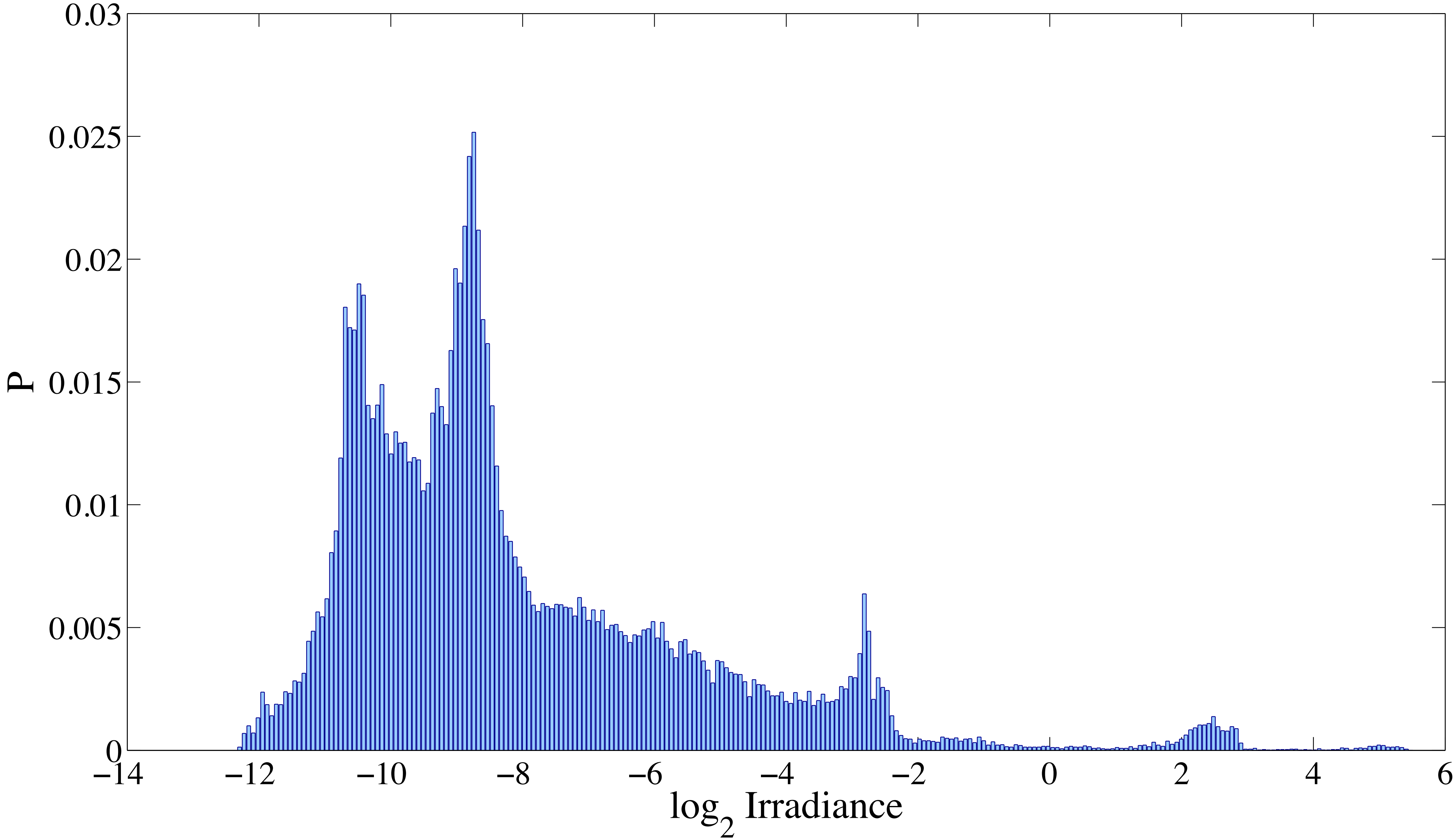
Exposure Metering: HDR Histogram



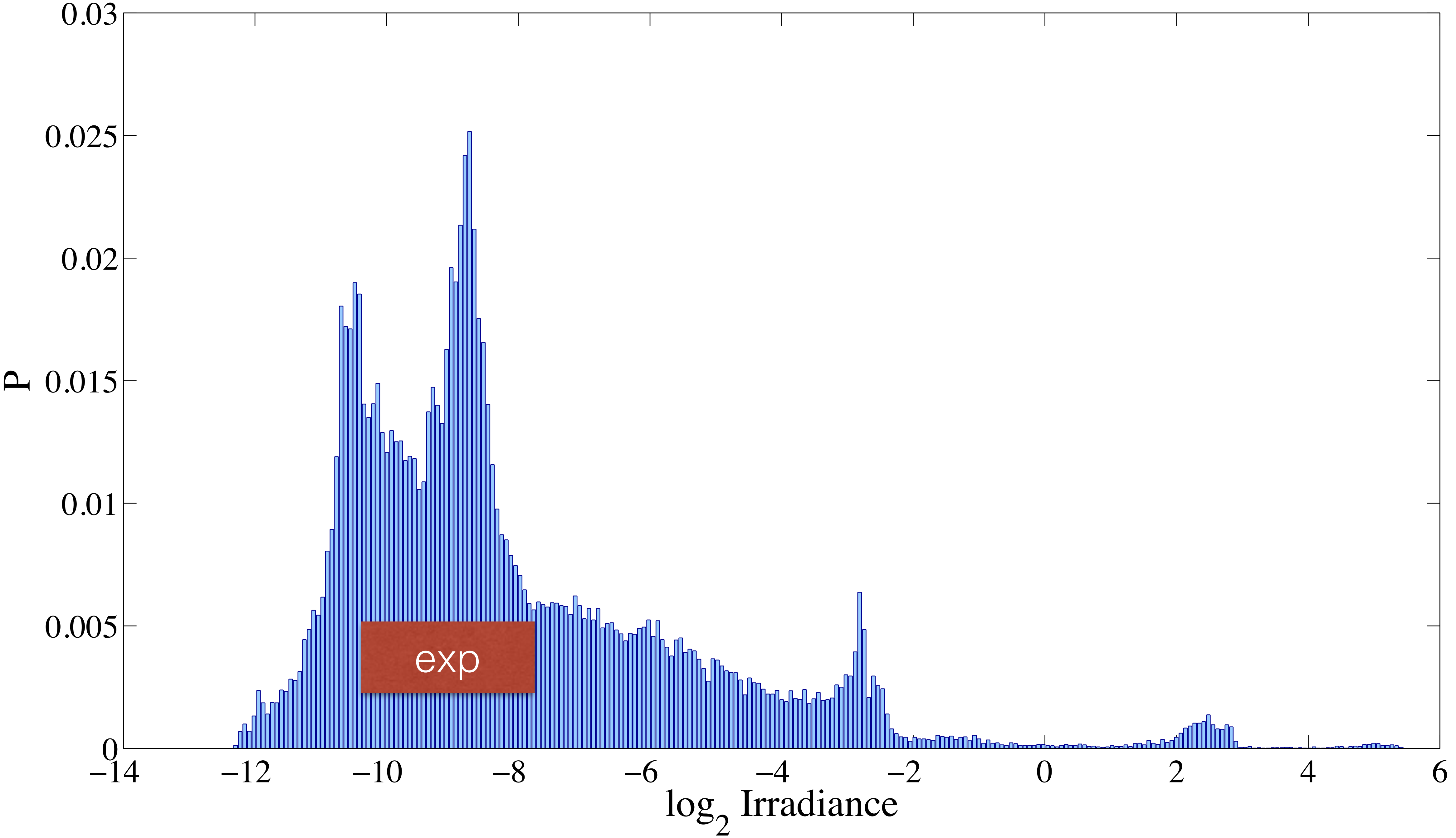
Exposure Metering

- Selection of exposure times based on:
    - HDR histogram
    - Noise model of the camera

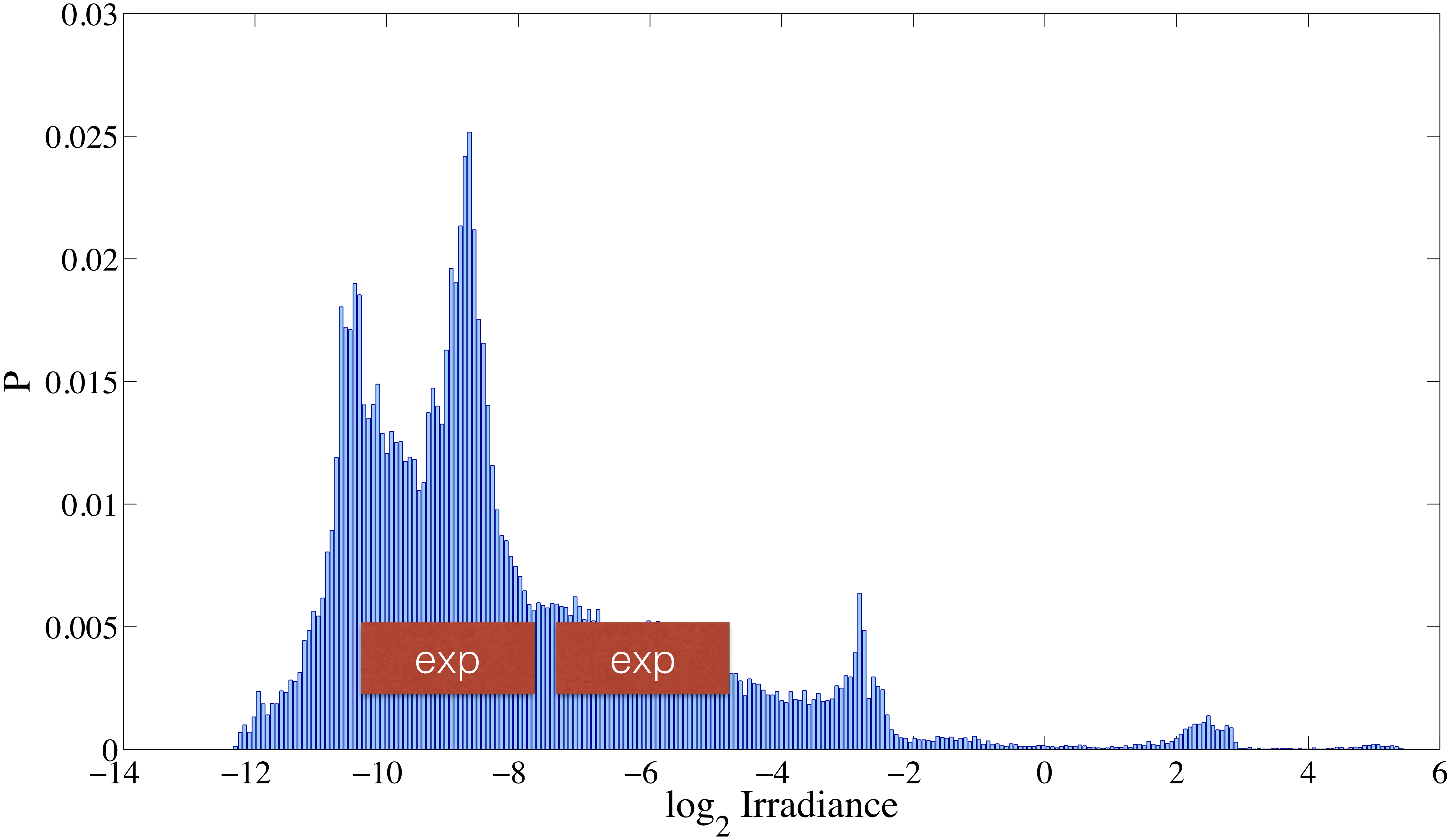
Exposure Metering: sampling



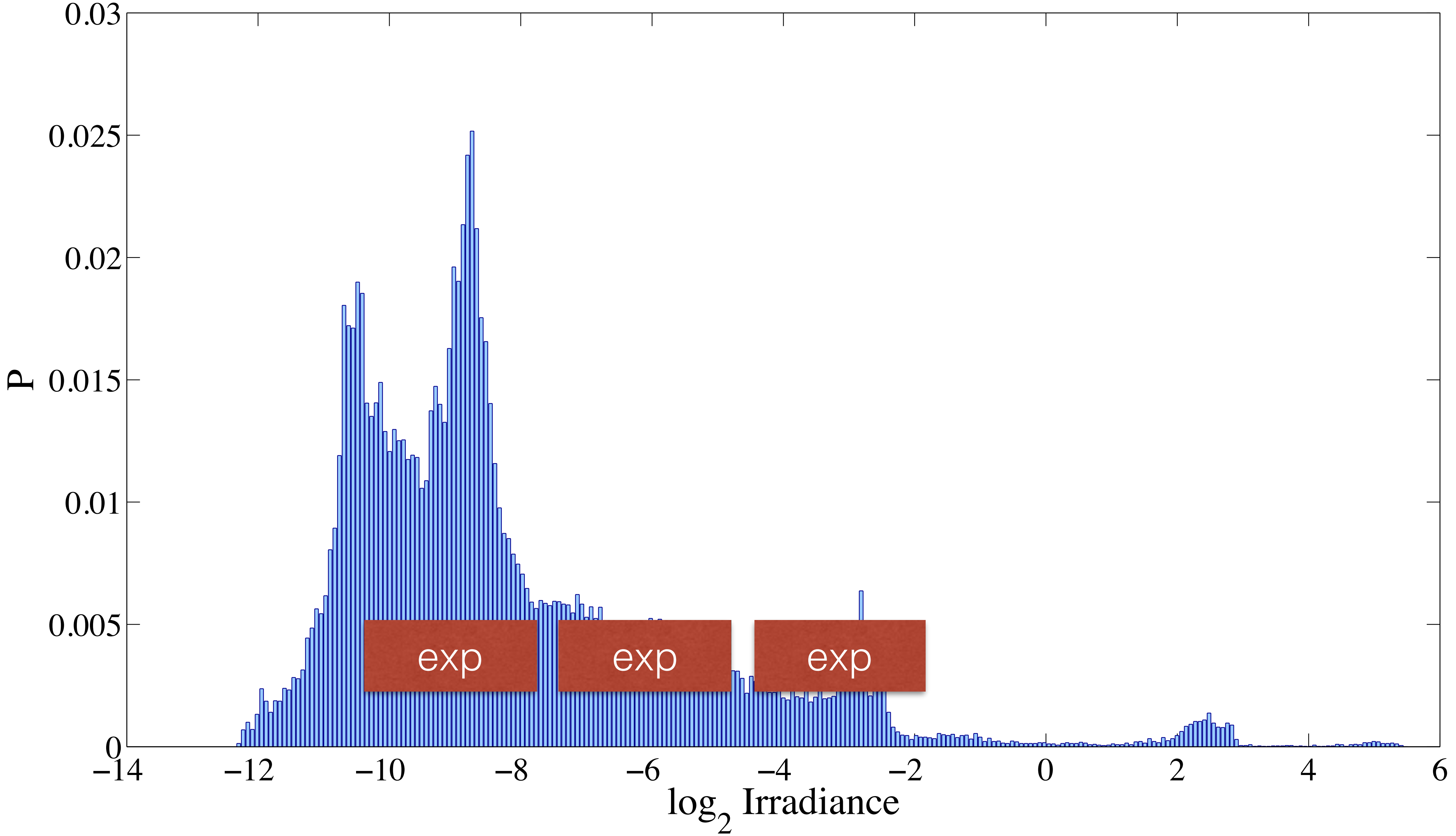
Exposure Metering: sampling



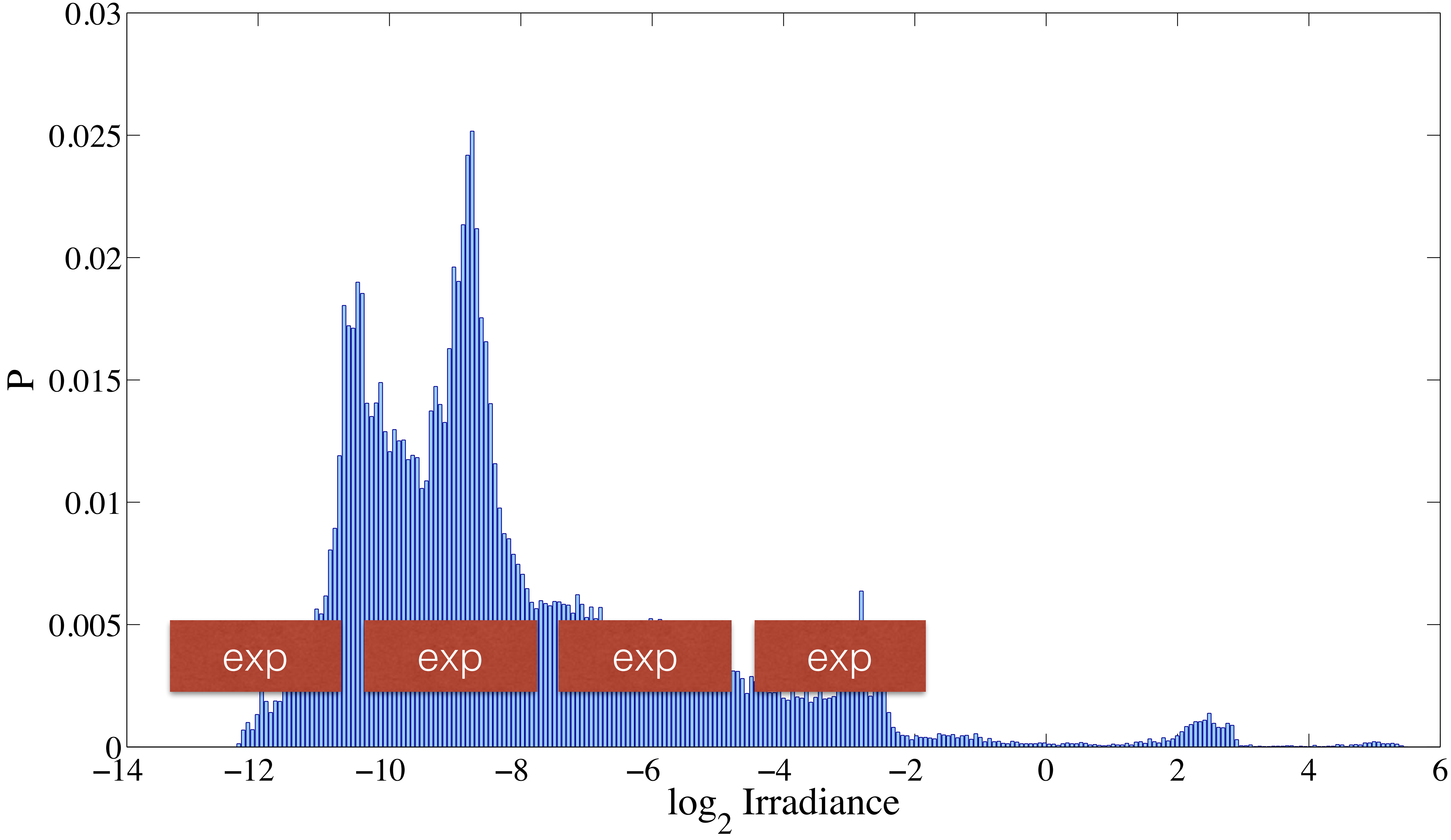
Exposure Metering: sampling



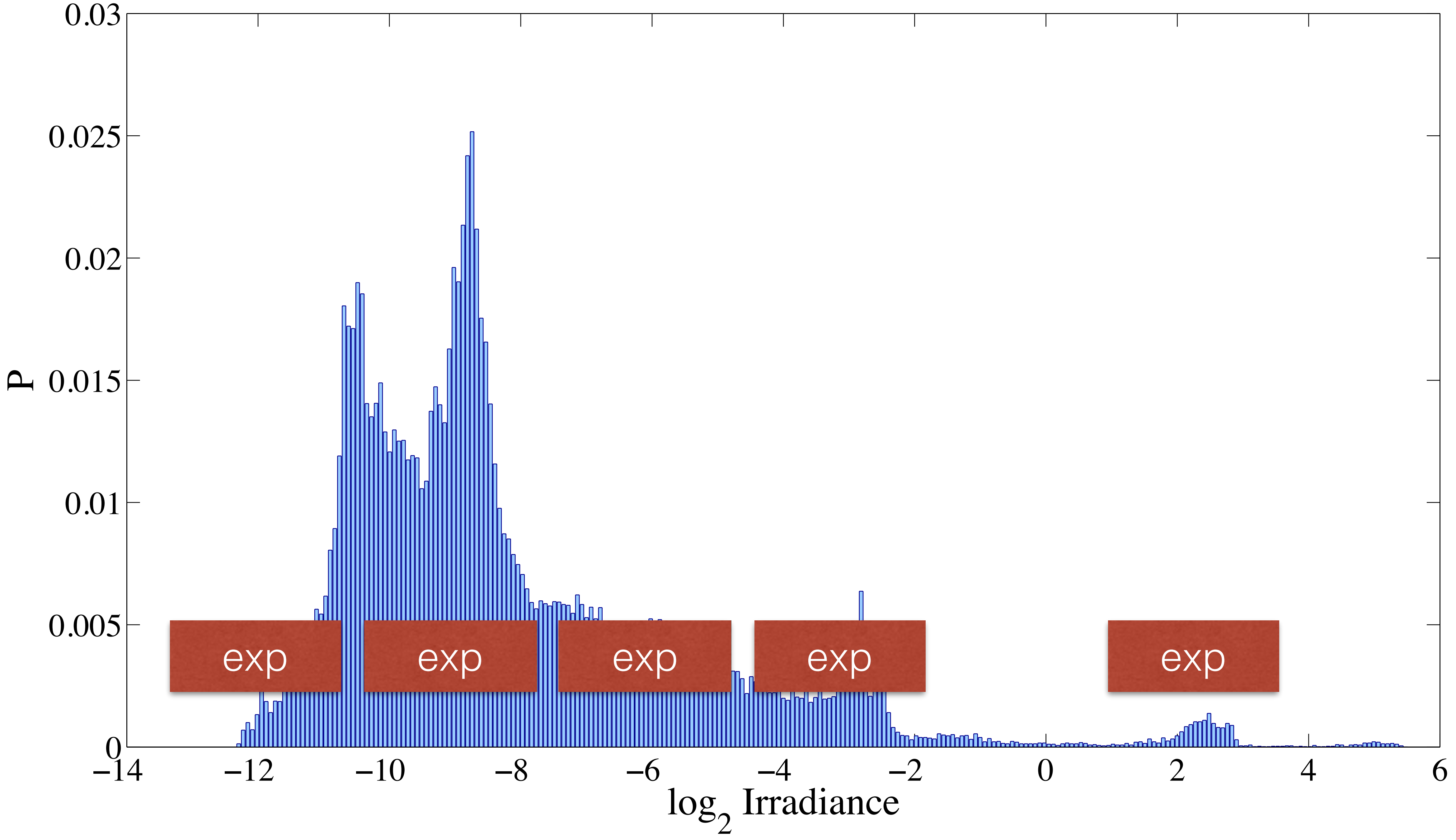
Exposure Metering: sampling



Exposure Metering: sampling



Exposure Metering: sampling



Linear Images

- What is a linear value?

$$I = a\,E$$

- where:
  - I the value recorded by the sensor
  - E is the radiance of the real value
  - a is a constant

Linear Images

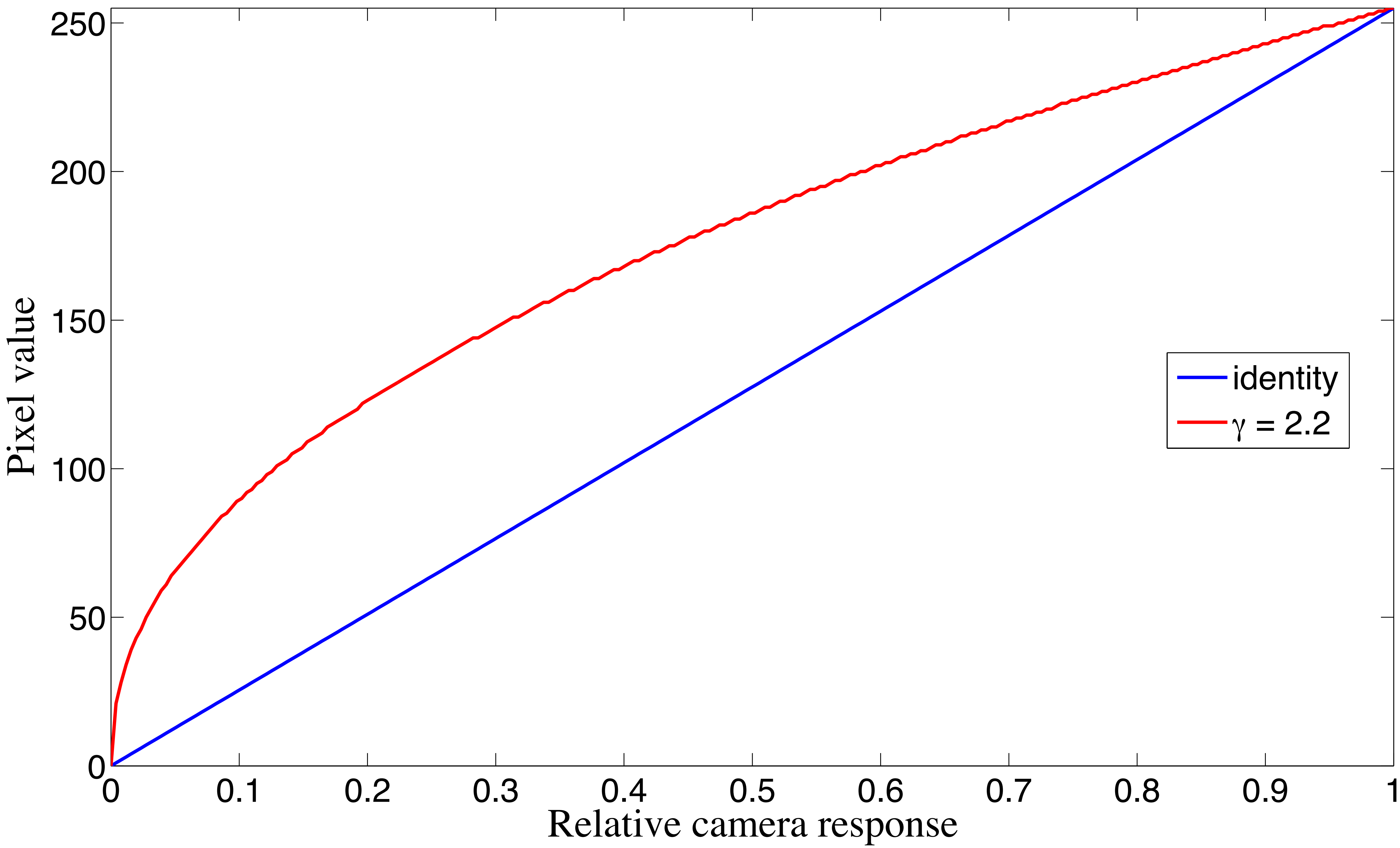
- High-end or prosumer camera can save RAW:
  - advantage: storing linear values
  - disadvantage: a lot of memory; no compression and 12-14bit per color channel

meanwhile in the real-word…

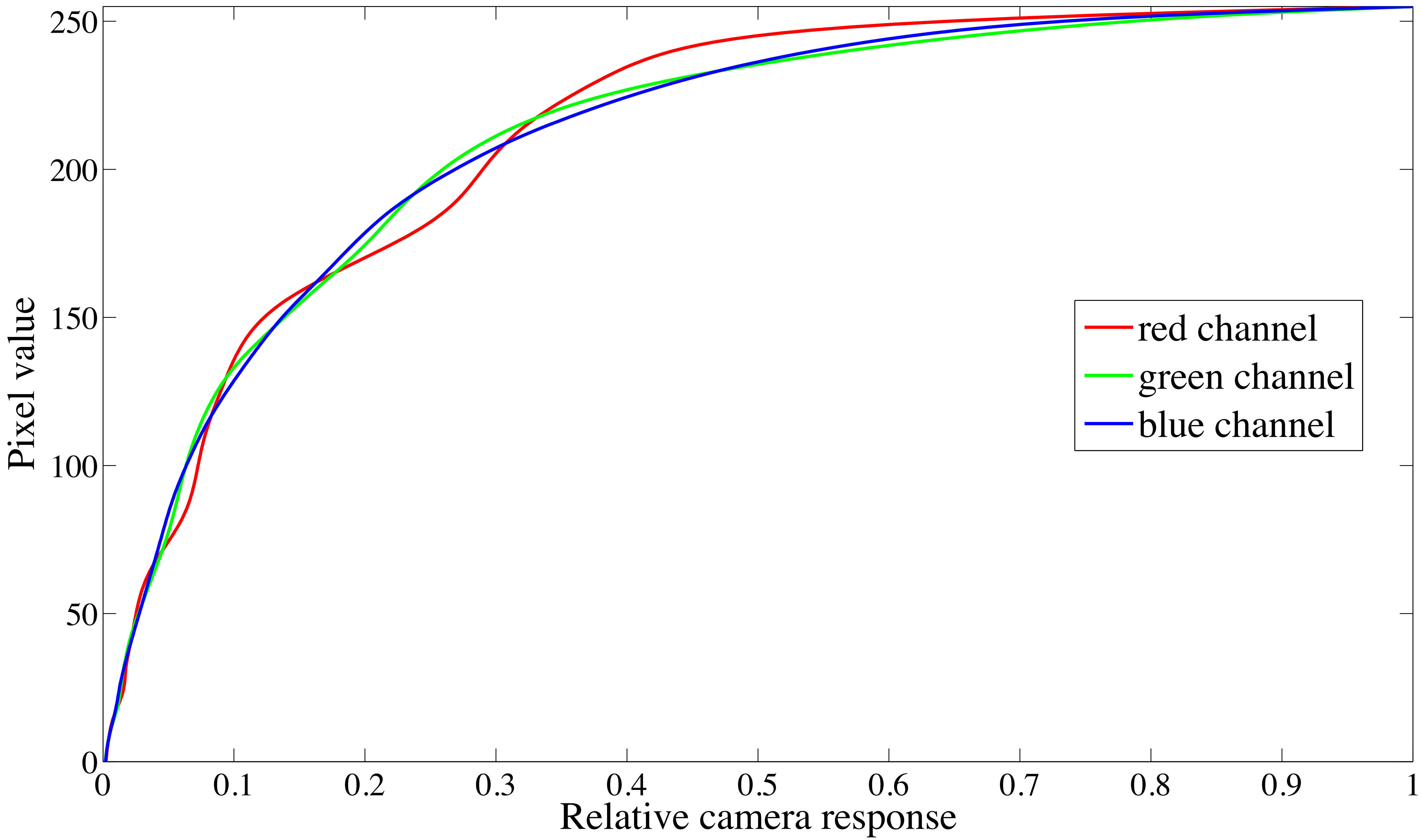
Linear Images

- Consumer cameras, smartphones, tables save typically JPEG at high quality (in the best case):
  - advantage: images are stored in little memory
  - disadvantage:
    - no linear values
    - images are stored applying an unknown function, f, called Camera Response Function (CRF)

Linear Images: example



Linear Images: example



Estimating CRF

- What can we do?
  - We can estimate the CRF or perform a radiometric calibration
- What can we do?
  - Taking a photograph with colorchecker and controlled environment
  - Taking photographs at different exposure times

Estimating CRF

$$Z = f(Et_i)$$

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

$$\ln f^{-1}(Z) = \ln E + \ln t_i$$

$$g(Z) = \ln E + \ln t_i$$

Estimating CRF

function to minimize

$$\mathcal{O} = \sum_{i=1}^{n}\sum_{j=1}^{M}\Big(w\big(Z_i(\mathbf{x}_j)\big)\big[g(Z_i(\mathbf{x}_j)) - \ln E(\mathbf{x}_j) - \ln t_i\big]\Big)^2$$

$$+ \lambda \sum_{x=T_{\min}+1}^{T_{\max}-1} \left(w(x)g''(x)\right)^2,$$

smoothing term

Estimating CRF

- To minimize the objective function, a ***dense*** linear system needs to be solved using SVD:
    - $(N_{exposures} \times N_{samples} + D + 1) \times (N_{samples} + D + 1)$
    - where D = 256 (discretization levels)
- We cannot use all pixels in the image:
    - too large system

Estimating CRF

- To minimize the objective function, a ***dense*** linear system needs to be solved using SVD
- We cannot use all pixels in the image:
  - too large system

Estimating CRF

- To minimize the objective function, a ***dense*** linear system needs to be solved using SVD
- We cannot use all pixels in the image:
    - too large system

Estimating CRF: samples selection

- **Idea1**: sampling in spatial domain

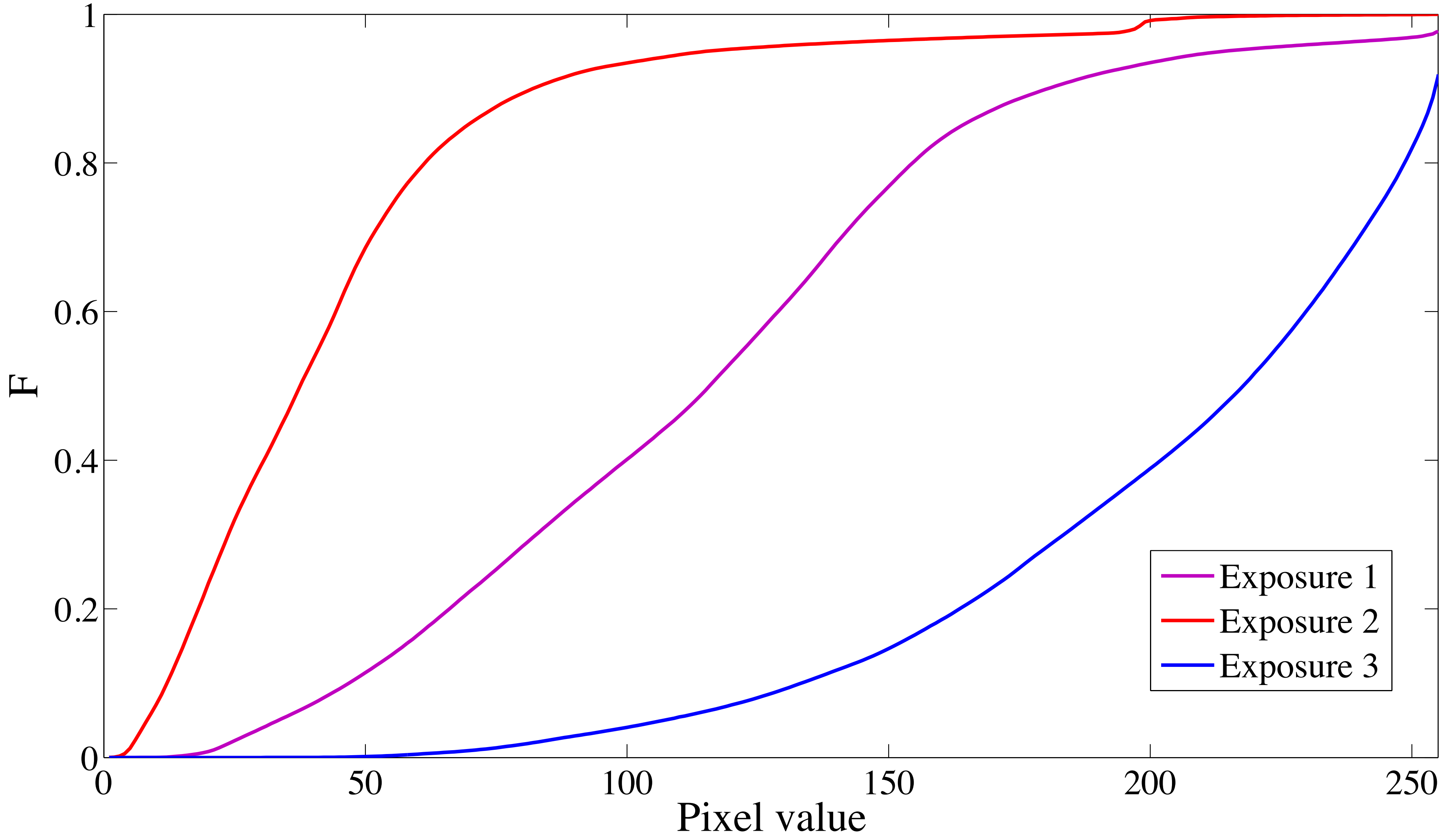
Estimating CRF: samples selection

- For each spatial sample (i, j)
  - Collect values at each exposure, Z_i, to obtain a sample vector:

$$[Z_0(i, j), ..., Z_n(i, j)]$$

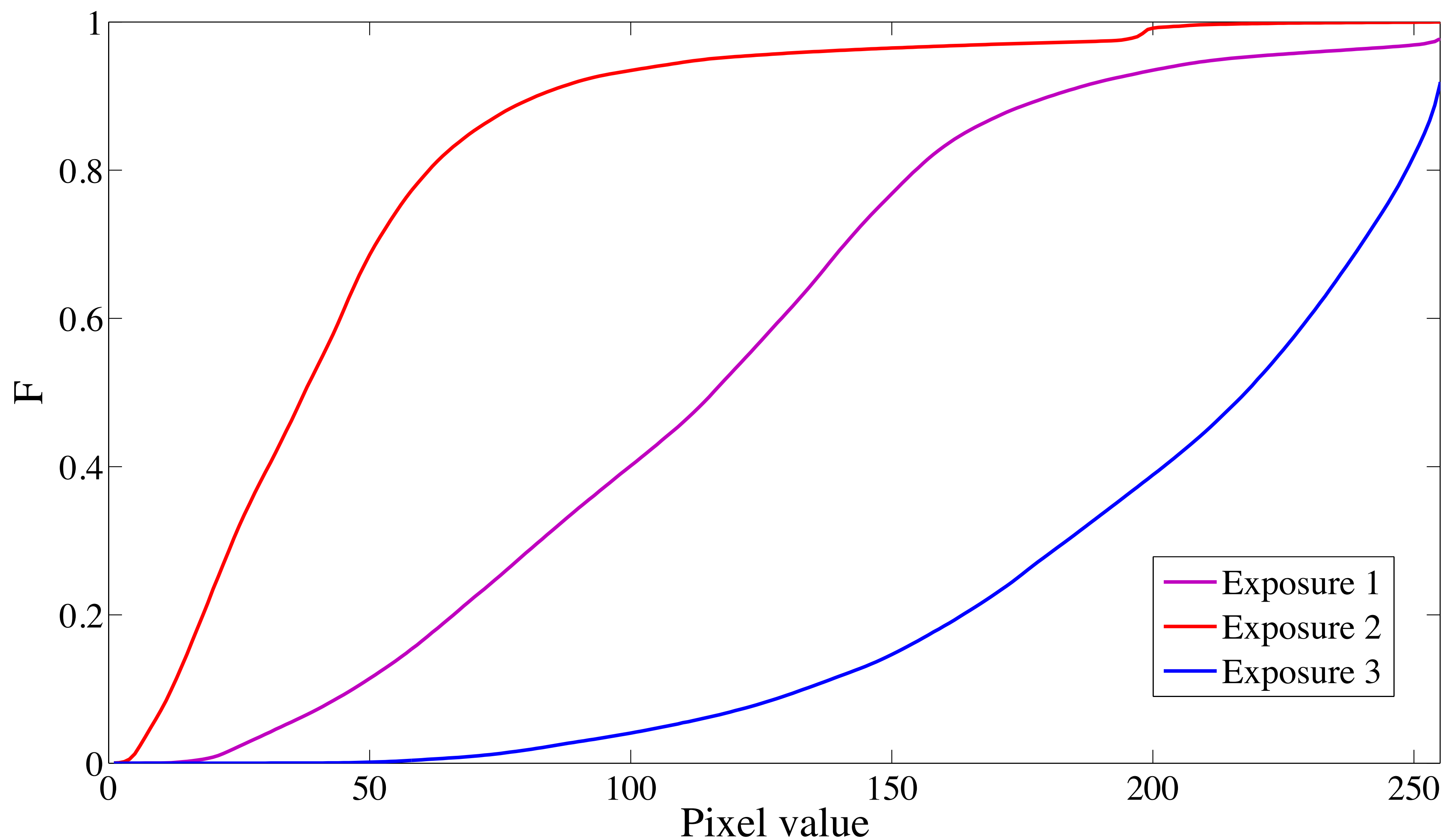
Estimating CRF: samples selection

- **Idea2**: in histogram domain, to randomly subsample the image



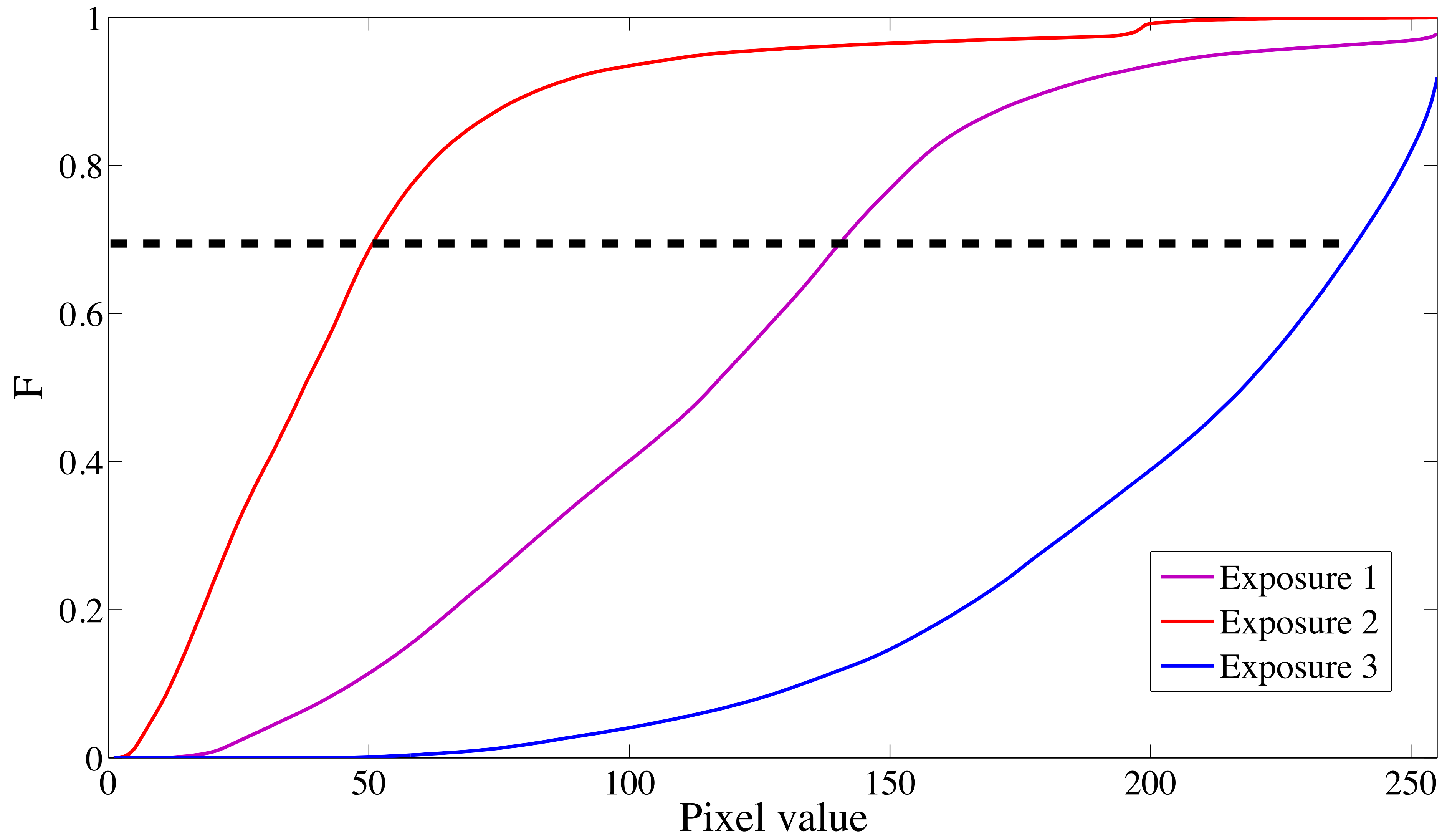
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



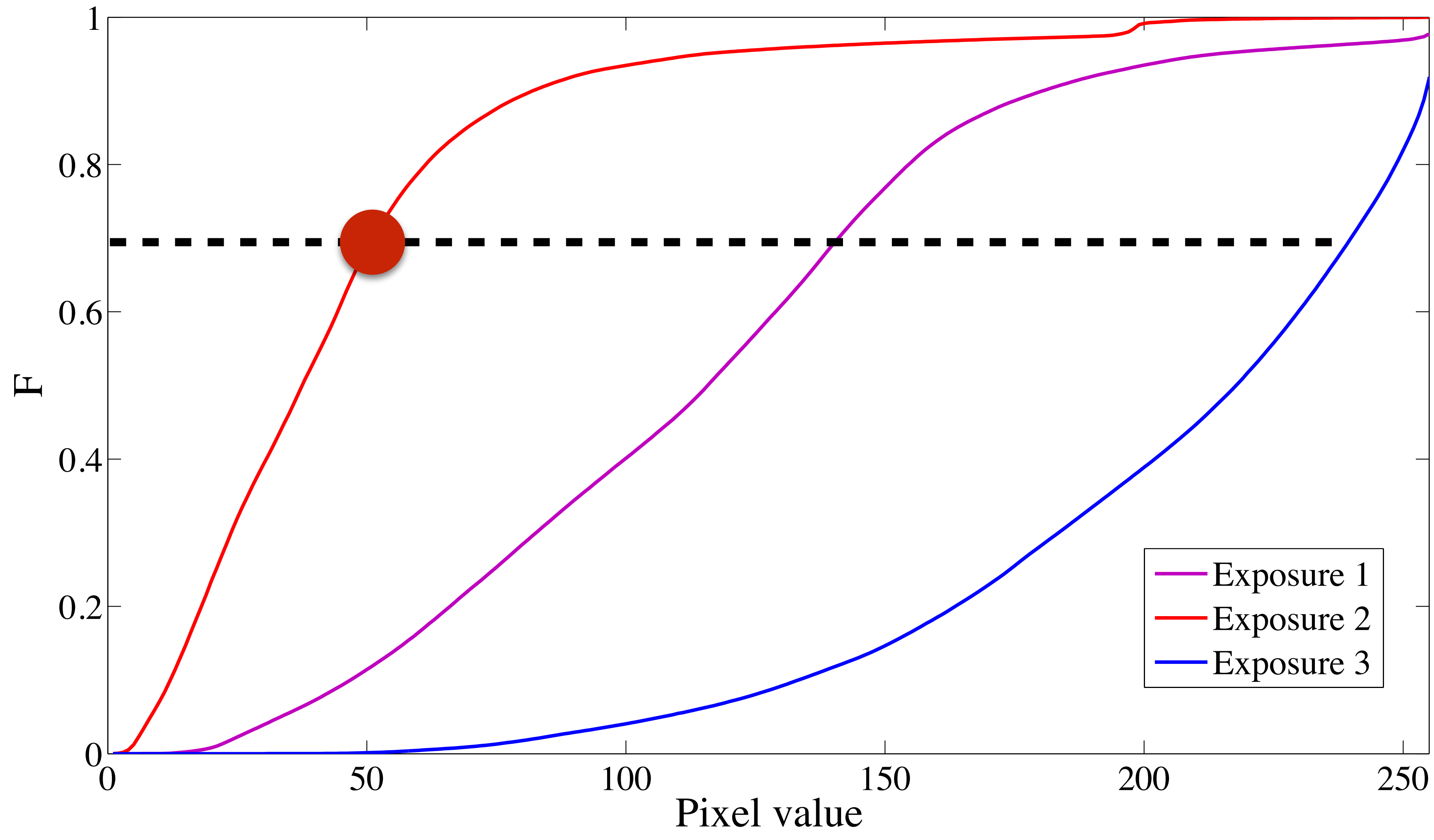
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



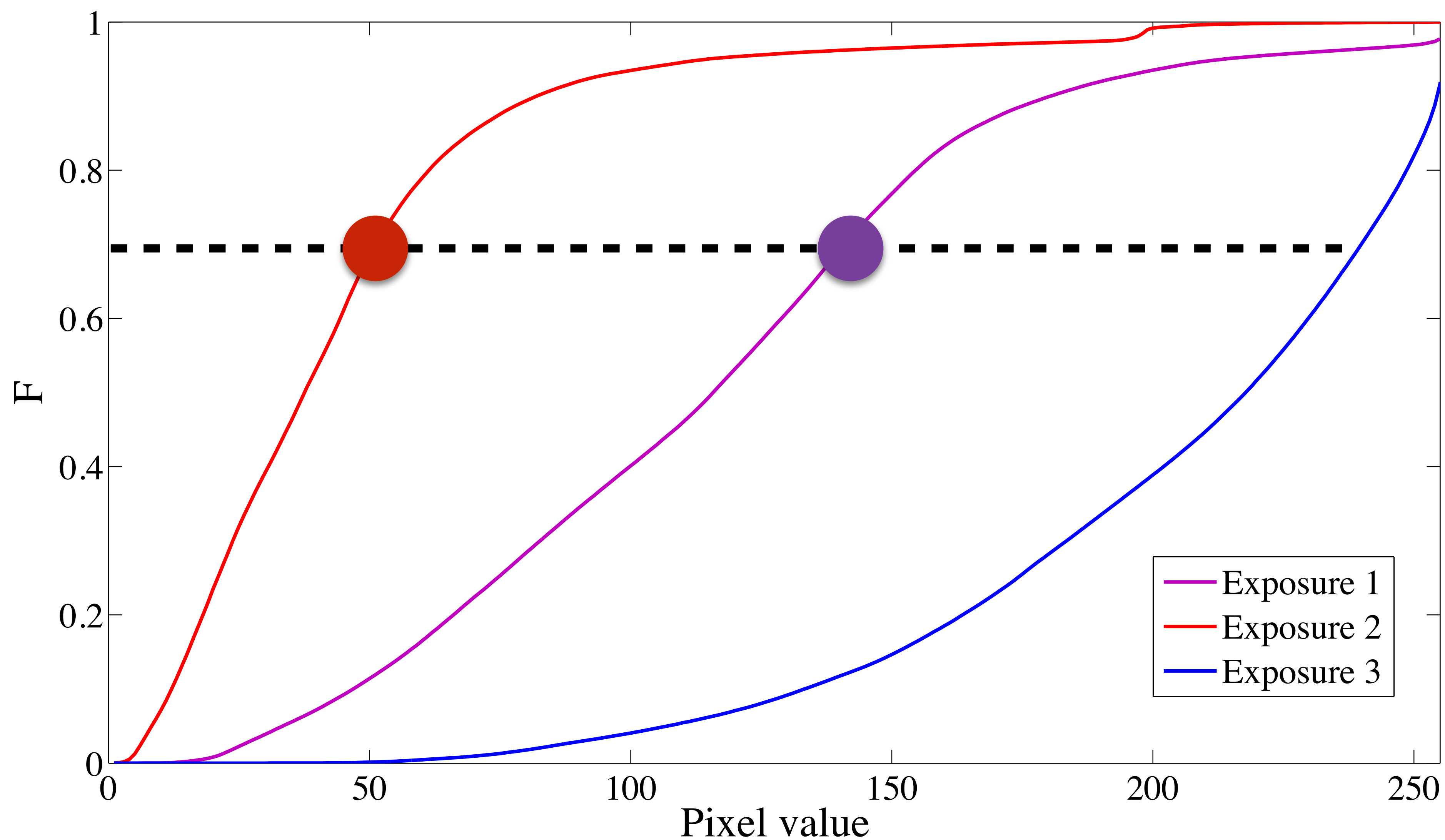
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



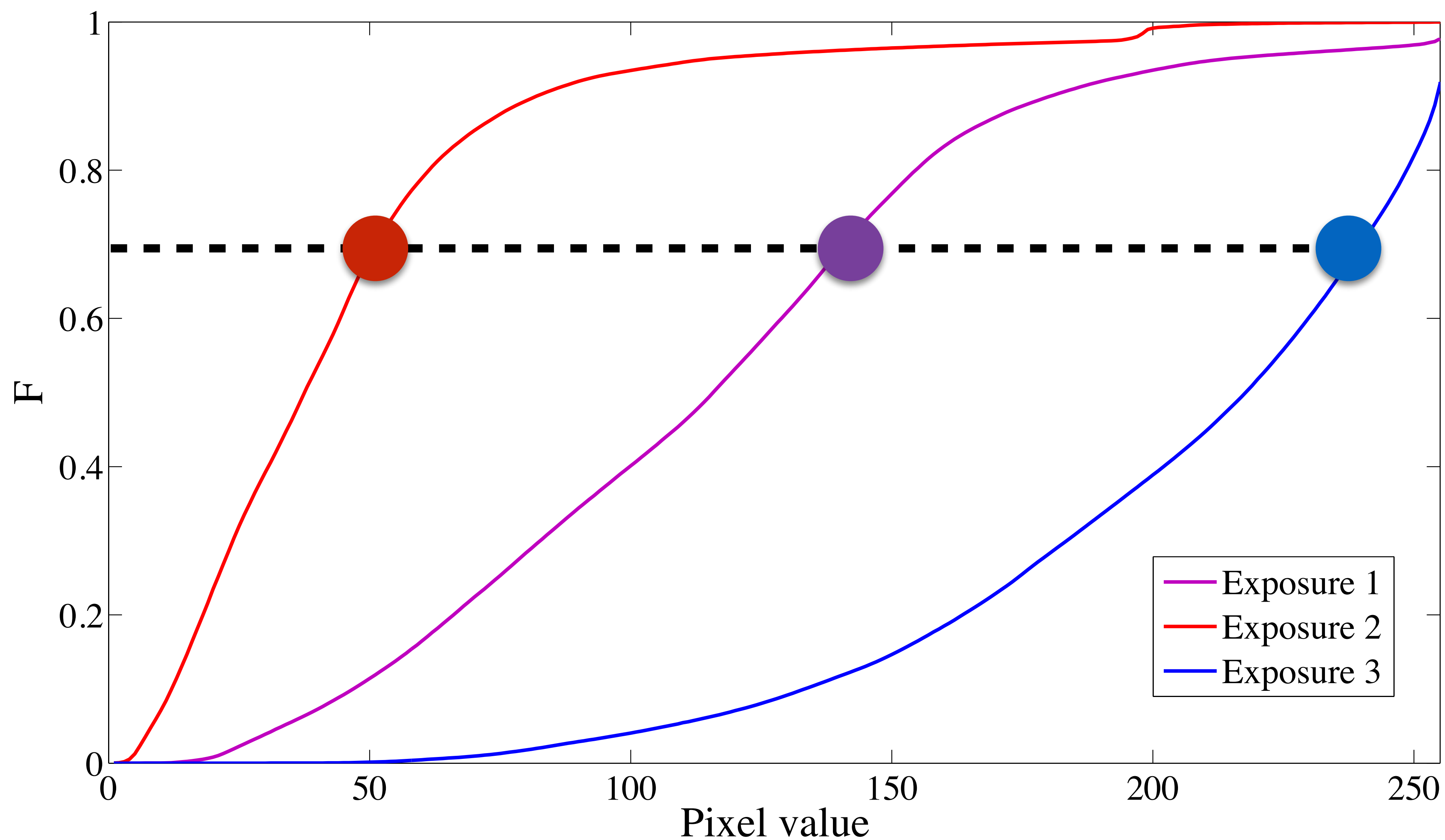
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



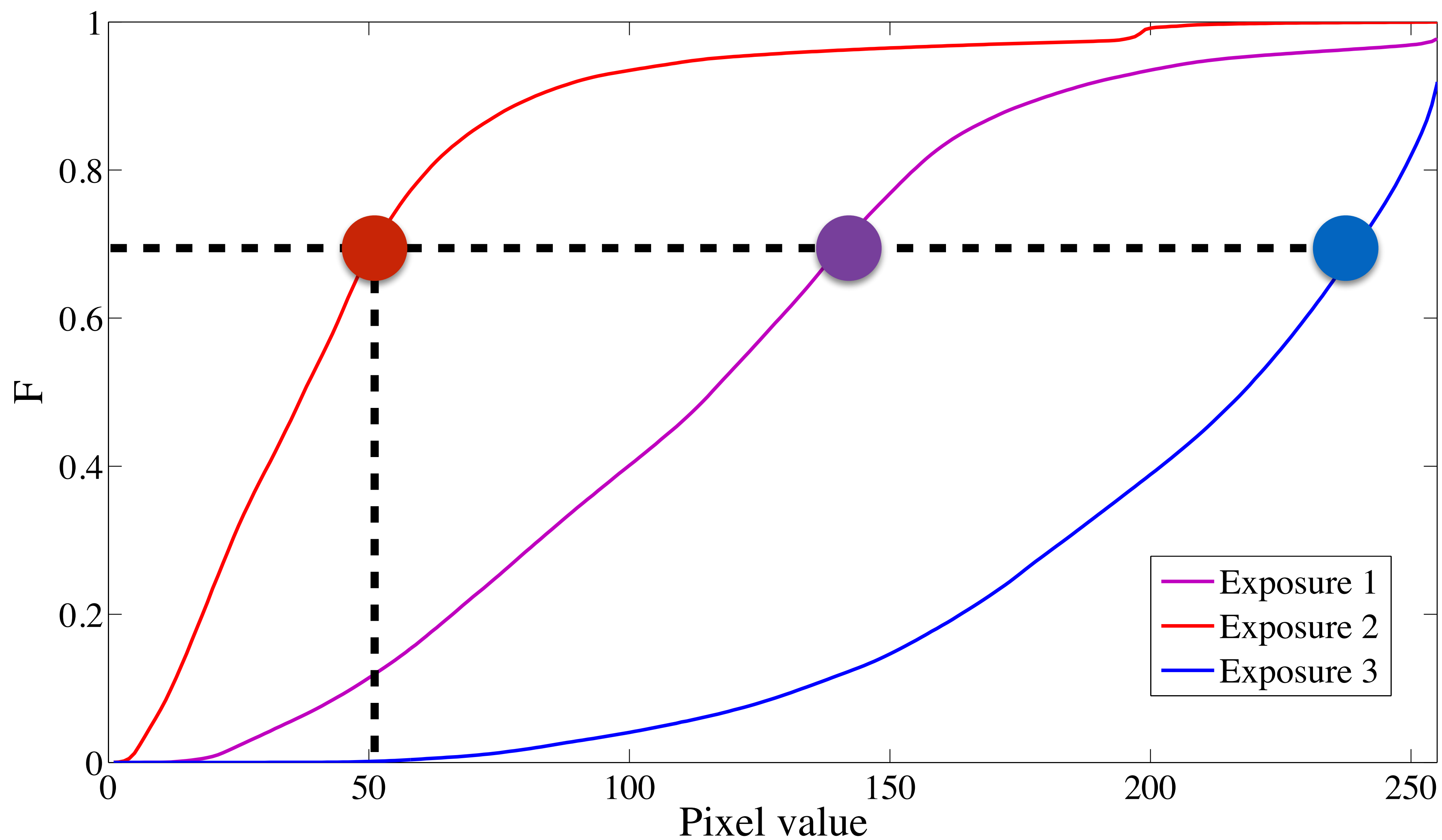
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



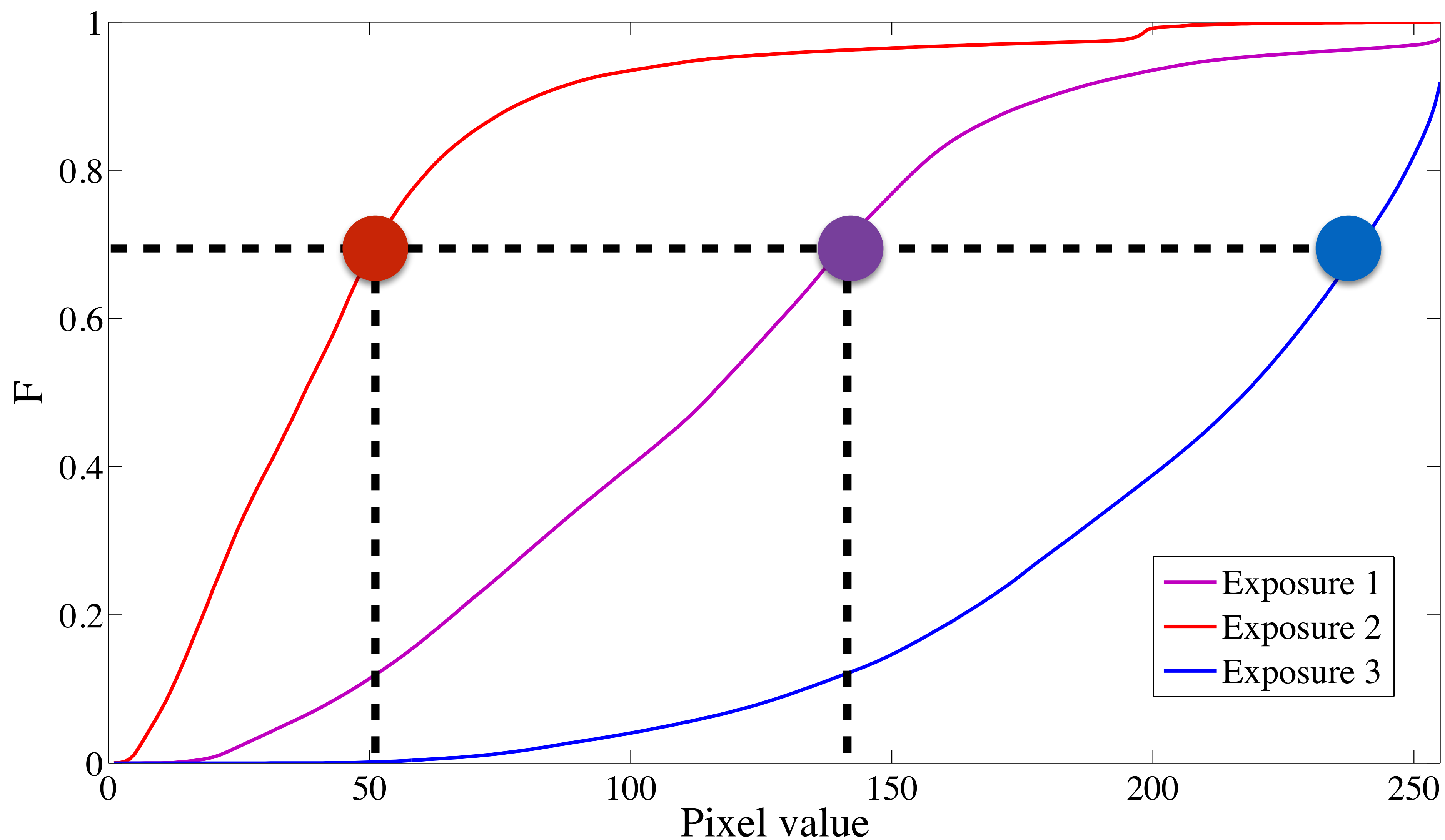
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



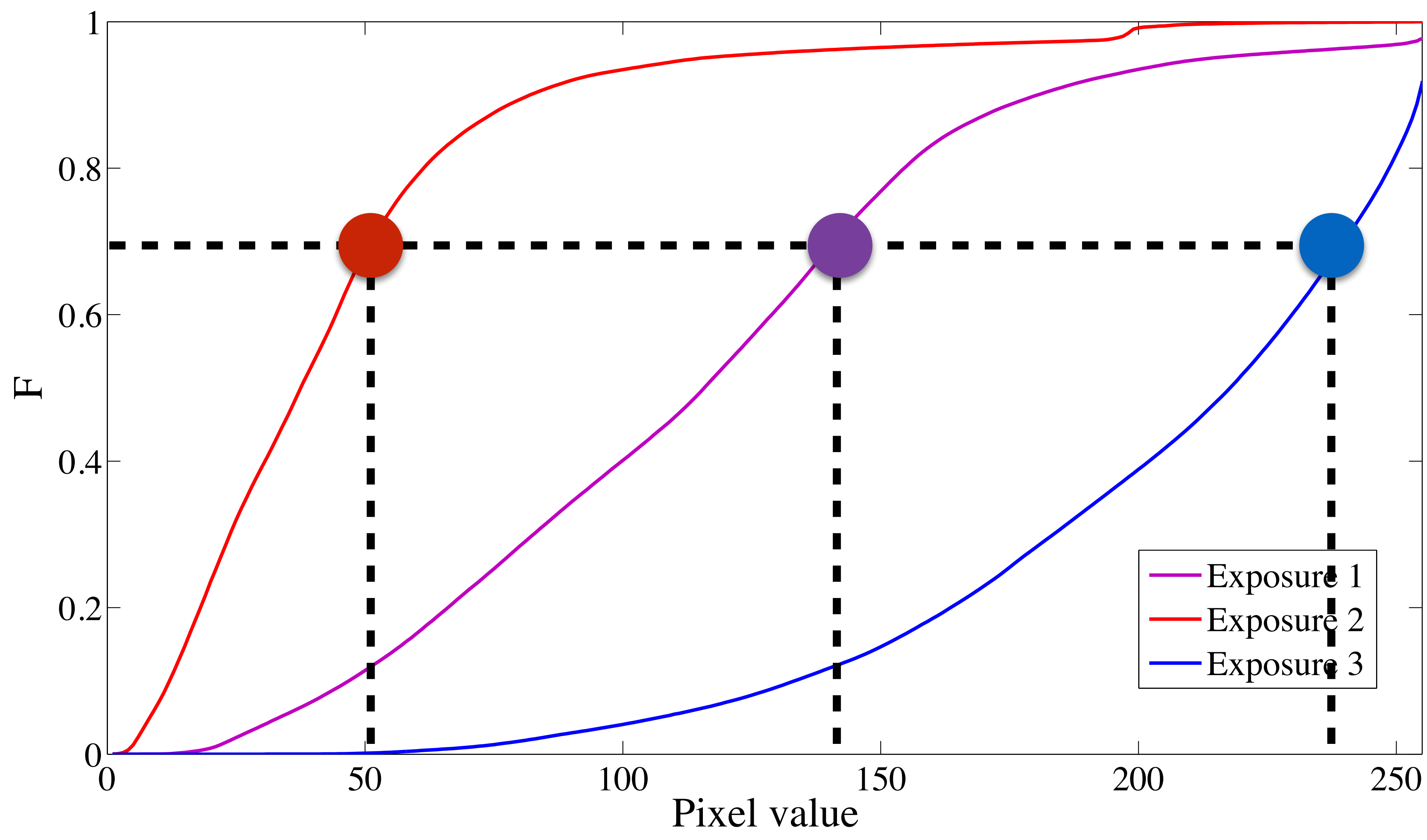
Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain



Estimating CRF: samples selection

- **Idea2**: sampling in histogram domain

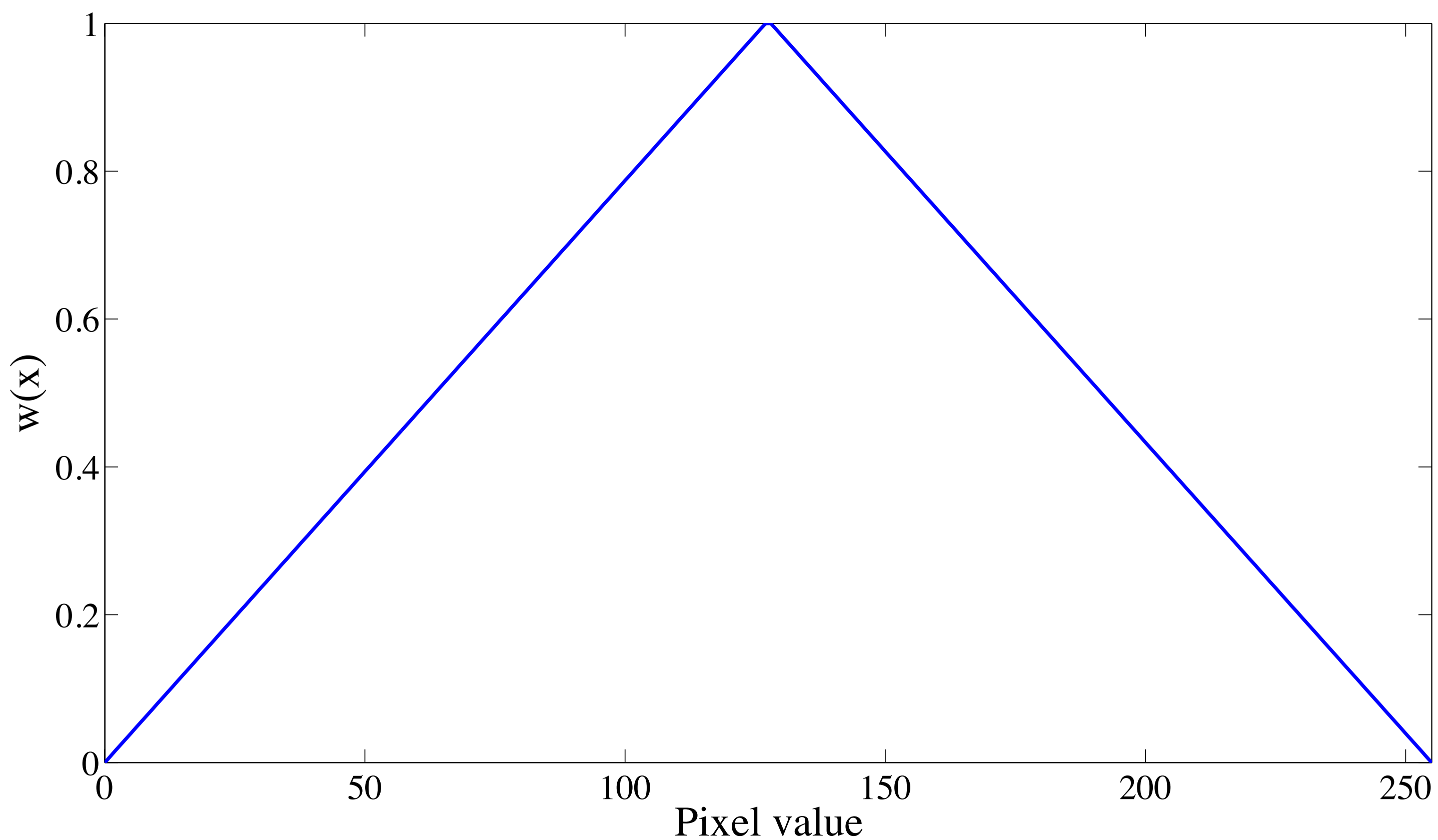


Estimating CRF: weighting function

- weighting function:
  - to avoid outliers during the estimate
  - shapes: tent, box with cut-off, Gaussian, etc.
  - outliers:
    - over-exposed pixels
    - under-exposed pixels

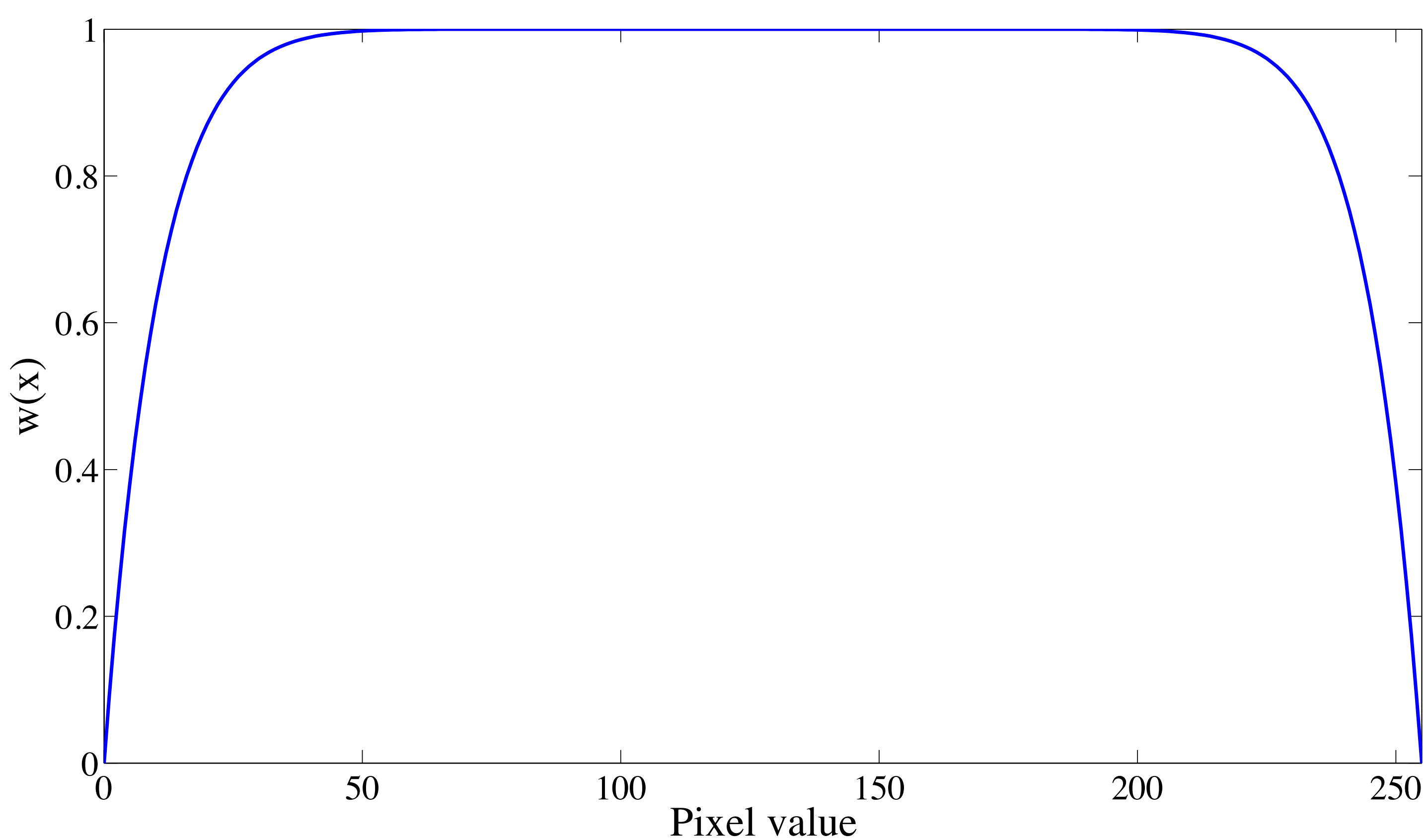
Estimating CRF: samples selection

$$w(x) = \begin{cases} x - \tau_{\min} & \text{if } x \leq \frac{1}{2}(\tau_{\max} + \tau_{\min}) \\ \tau_{\max} - x & \text{if } x > \frac{1}{2}(\tau_{\max} + \tau_{\min}) \end{cases} \quad \text{with } x \in [0, 255]$$



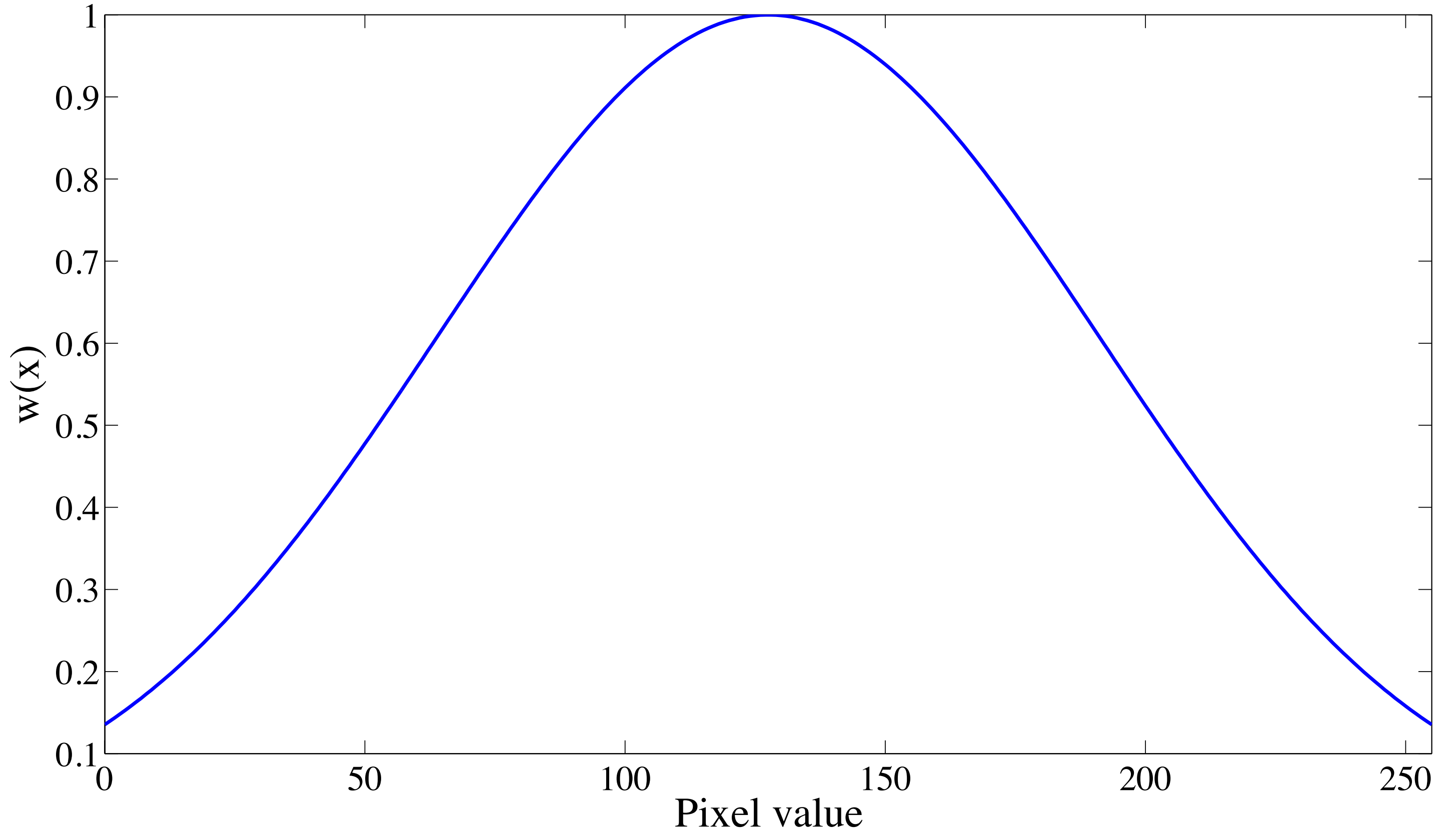
Estimating CRF: samples selection

$$w(x) = 1 - (2x - 1)^{12}$$



Estimating CRF: samples selection

$$w(x) = e^{-4\frac{(x-0.5)^2}{2(0.5)^2}}$$



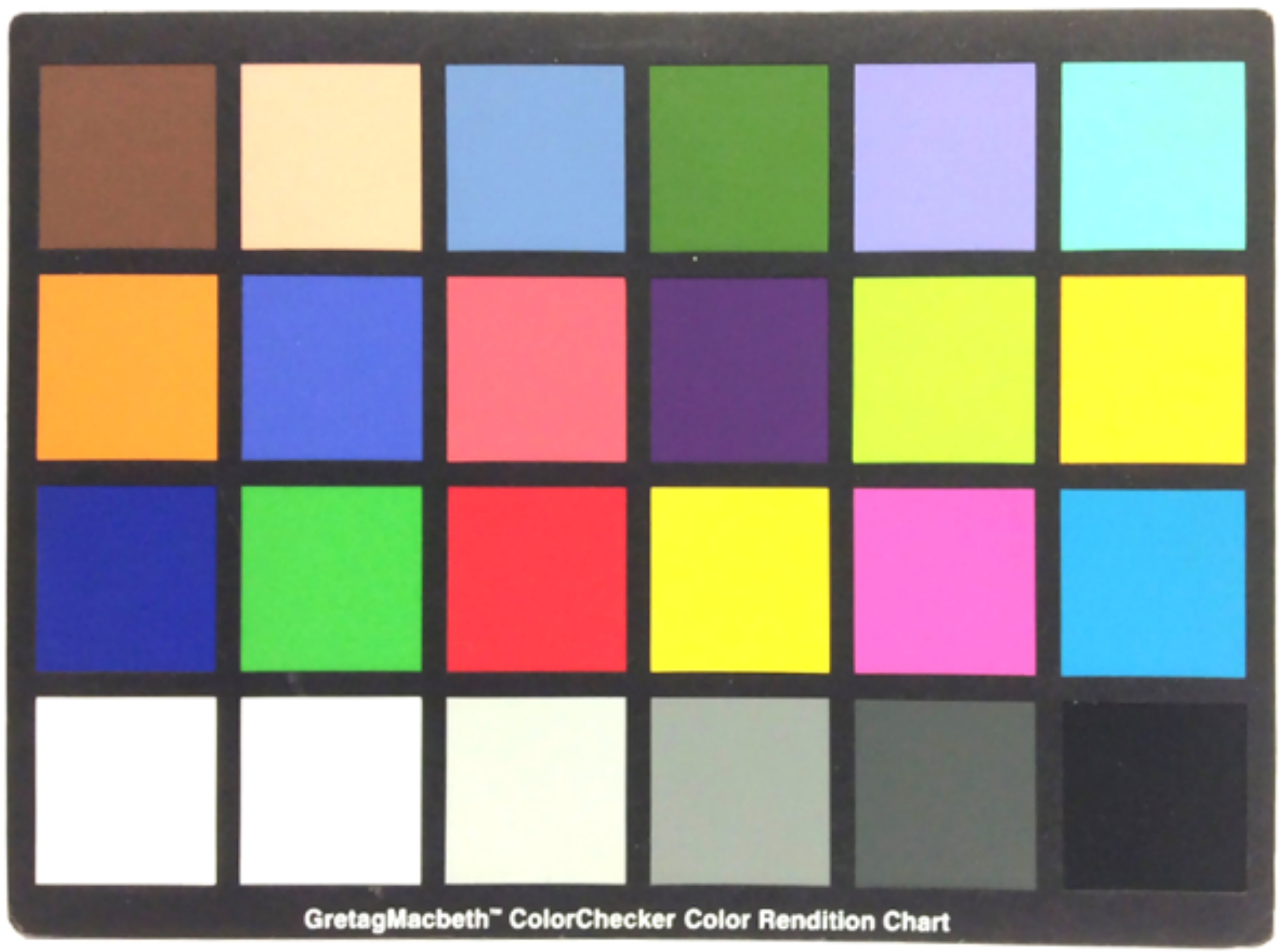
Estimating CRF: samples selection

- Other methods?
  - To fit a N-dimensional polynomial

$$f(x) = \sum_{i=0}^{N} c_i x^i$$

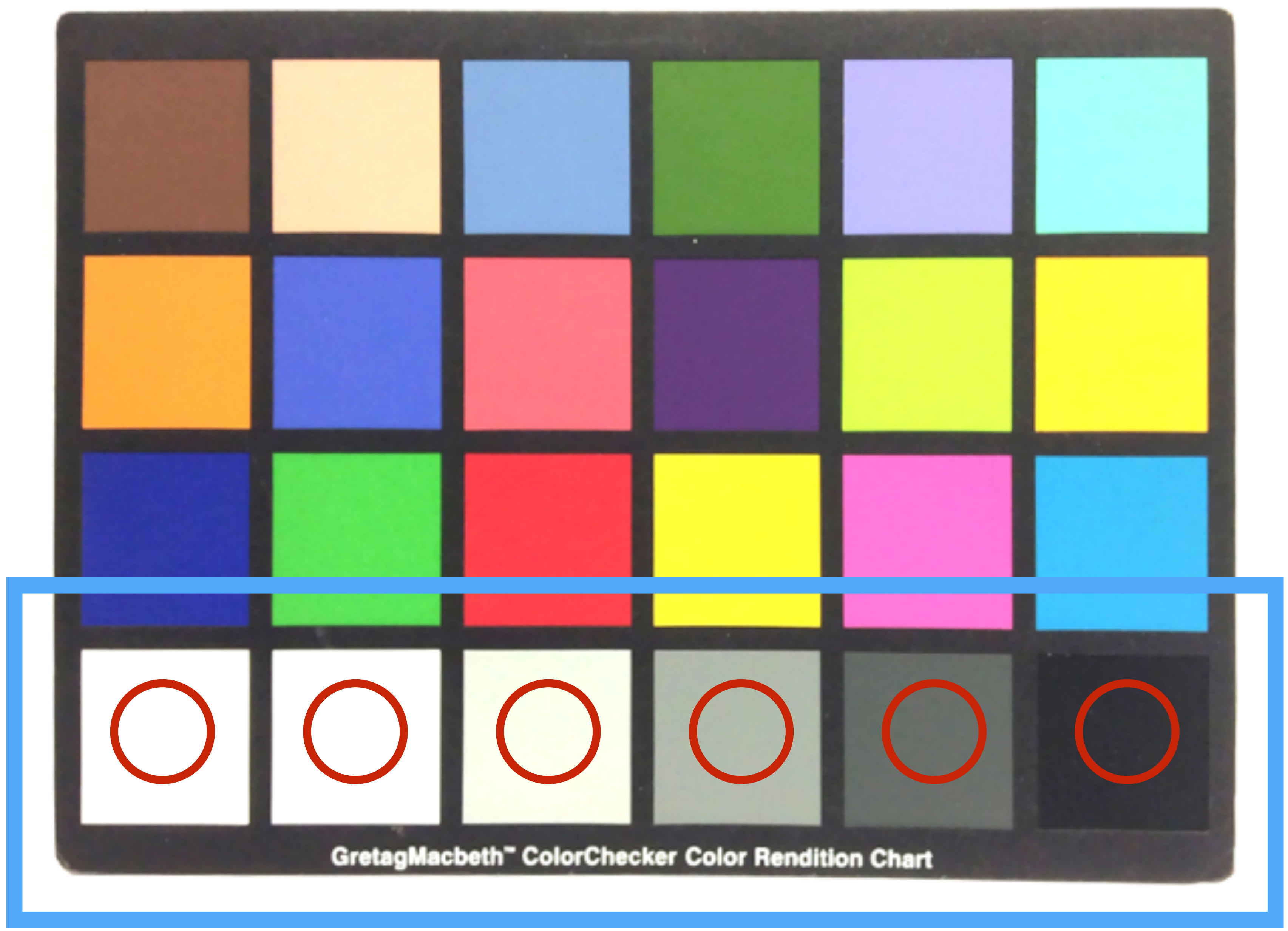
  - How to chose N? Brute force: trying different fits, from N=1 to N=10 and chose the one with the smallest error

Estimating CRF: colorchecker based

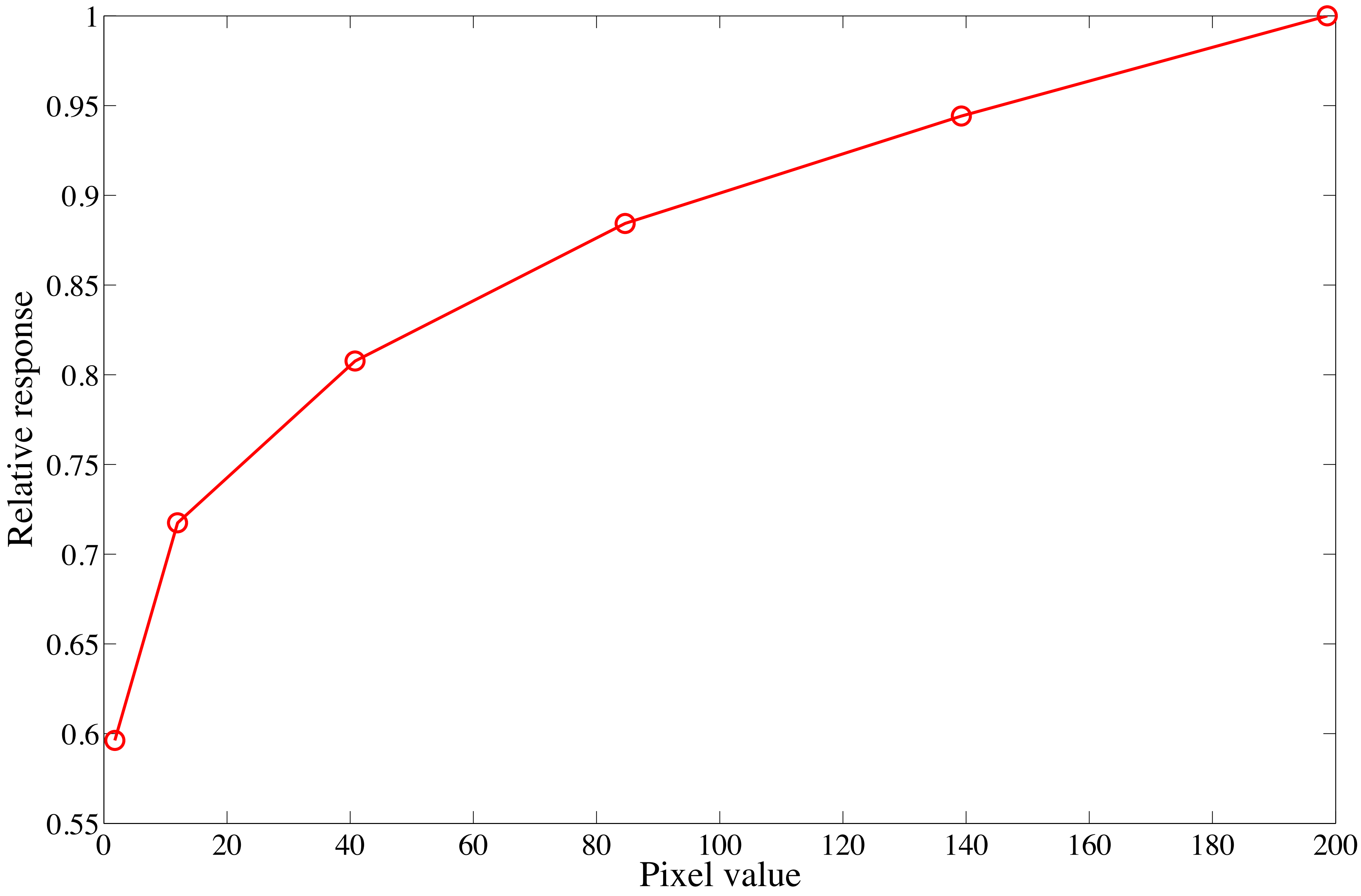


Estimating CRF: colorchecker based

Estimating CRF: colorchecker based



Estimating CRF: colorchecker based



Estimating CRF: colorchecker based

- This method is computationally cheap, and it offers a ground truth but:
  - Color checker
  - Luminance meter or photometer
  - Better to have controlled lighting
  - Few points… interpolation

Where are we?

- We know how to capture enough images
- We know how to compute the CRF
- We need to build the HDR image from the LDR ones

HDR merge

$$E(\mathbf{x}) = \frac{\sum_{i=1}^{n} \frac{1}{t_i} w(Z_i(\mathbf{x})) f^{-1}(Z_i(\mathbf{x}))}{\sum_{i=1}^{n} w(Z_i(\mathbf{x}))}$$

HDR merge: noise reduction

$$E(\mathbf{x}) = \frac{\sum_{i=1}^{n} w(Z_i(\mathbf{x}))t_i^2 \frac{f^{-1}(Z_i(\mathbf{x}))}{t_i}}{\sum_{i=1}^{n} w(Z_i(\mathbf{x}))t_i^2}$$

Note: this gives more weight to long-exposure images (less noise) than short-exposure images (more noise)

Exposure time

- Exposure time how is it computed?
- Typically using shutter speed, but we need to take into account of:
    - ISO
    - Aperture

Exposure time

- Keeping shutter and ISO constant, and varying the aperture the image gets brighter or darker:

F/8

F/5.6

F/4

Exposure time

- Keeping shutter speed and aperture constant, and varying the ISO the image gets brighter or darker:

ISO 200

ISO 400

ISO 800

Exposure time

$$t_i^e = \frac{I t_i}{K A^2}$$

- I is the ISO value
- A is the aperture value
- t_i is the shutter speed (time)
- K is a camera manufacturer constant in [10.6, 13.4]

Example

t = 1/128s

t = 1/32s

t = 1/8s

Example

t = 1/128s

t = 1/32s

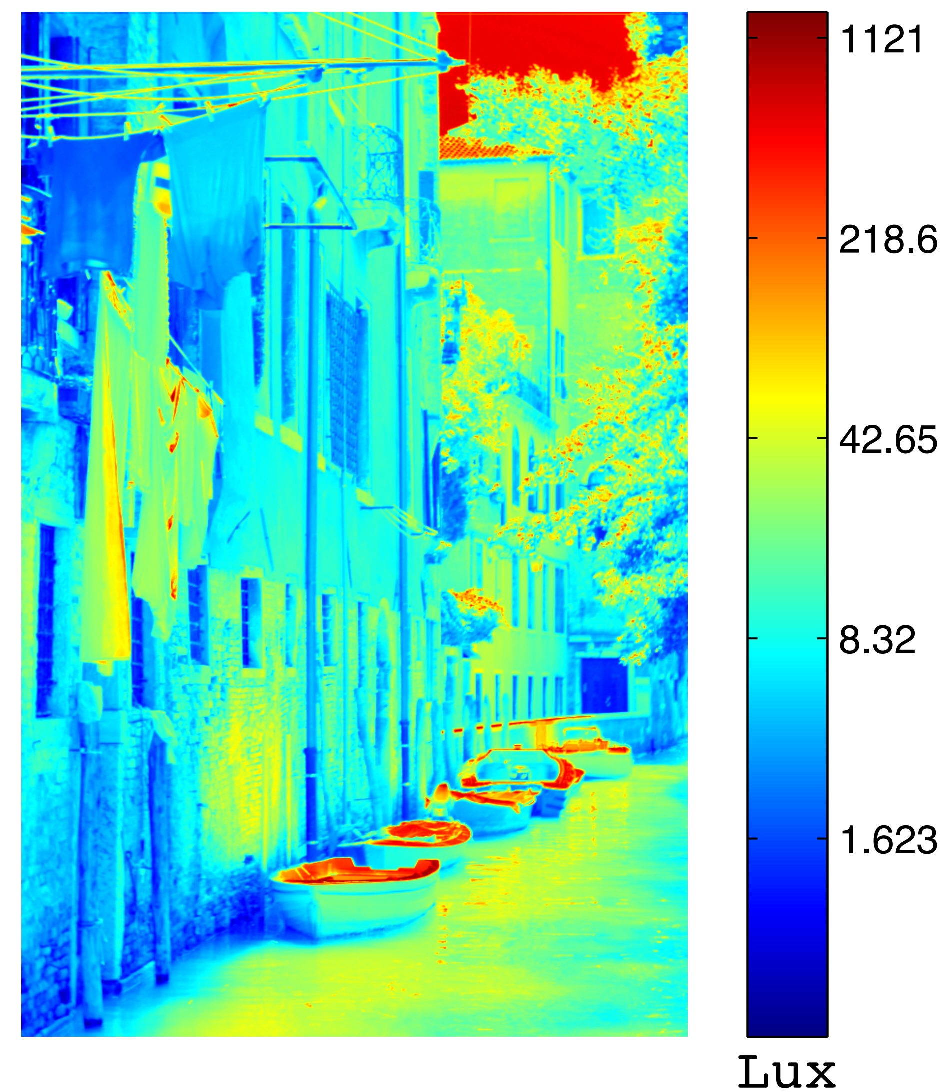
t = 1/8s

Example

t = 1/128s

t = 1/32s

t = 1/8s



HDR Formats

- Once, an HDR image is merged, it has to be stored
- 8-bit unsigned char encoding per color channels is not enough —> limited range [0,255]
- The range of values for natural scenes can be very large —> [10^{-7} 10^{9}] cd/m^2

HDR Formats: floating point

- Typically, HDR pixels are stored using 32-bit floating point numbers per color channel:

- This means four times the amount of memory for an uncompressed LDR pixel!

- Moreover, IEEE 754 encoding is a bit wasted, more values that what is needed

HDR Formats: RGBE

- **Idea**: red, green, and blue color channel for a given error may have a very similar exponent, only mantissa is changing!
- A standard integrated in some OS, e.g. OS X
- It can not encode negative values

HDR Formats: RGBE

$$E_m = \left\lceil \log_2 \max(R, G, B) + 128 \right\rceil$$

$$R_m = \left\lfloor \frac{256R}{2^{E_m - 128}} \right\rfloor$$

$$G_m = \left\lfloor \frac{256G}{2^{E_m - 128}} \right\rfloor$$

$$B_m = \left\lfloor \frac{256B}{2^{E_m - 128}} \right\rfloor$$

R_m	G_m	B_m	E_m	32-bit
8-bit	8-bit	8-bit	8-bit	

HDR Formats: LogLuv

- **Idea**: convert RGB colors in the LogLuv color space; colors require less precision than intensity values
- **Advantage**: intensity and color values are separated good for post-processing
- **Two versions**: 24-bit and 32-bit

HDR Formats: LogLuv

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathbf{M}_{RGB \to XYZ} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{X}{X+Y+Z} \\ \frac{Y}{X+Y+Z} \end{bmatrix}$$

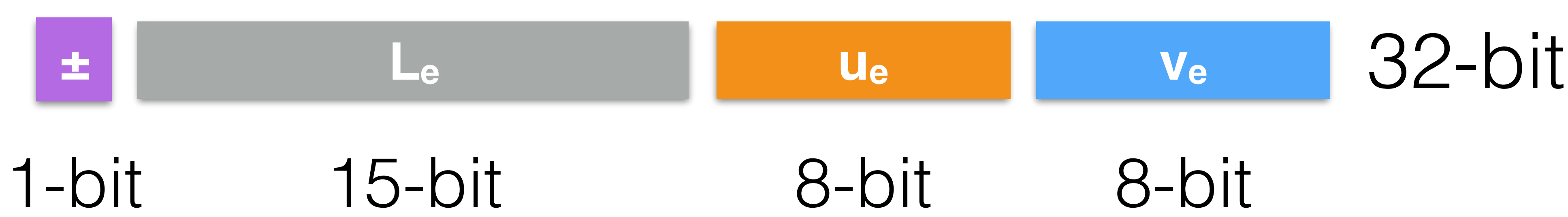
$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} \frac{4x}{-2x+12y+3} \\ \frac{4x}{-2x+12y+3} \end{bmatrix}$$

HDR Formats: LogLuv 32-bit

$$L_e = \lfloor (256 \log_2 Y + 64) \rfloor$$

$$u_e = \lfloor 410u' \rfloor$$

$$v_e = \lfloor 410v' \rfloor$$



HDR Formats: LogLuv 24-bit

$$L_e = \lfloor (64 \log_2 Y + 12) \rfloor$$

$$u_e = \lfloor 410u' \rfloor$$

$$v_e = \lfloor 410v' \rfloor$$

±	L_e	u_e	v_e	24-bit
1-bit	10-bit	7-bit	7-bit	

HDR Formats: OpenEXR

- Standard de facto for HDR “digital negative” values
- Proposed by ILM in 2002 as a digital negative for movies and CGI productions
- Half format (16-bit) for each color channel:
  - Dynamic range: [0.000061 , 65504]
- OpenSource on github:
  - https://github.com/openexr/openexr

HDR Formats: OpenEXR

$$H = \begin{cases} 0 & \text{if } \big(M = 0 \wedge E = 0\big), \\ (-1)^S 2^{E-15} + \frac{M}{1024} & \text{if } E = 0, \\ (-1)^S 2^{E-15}\left(1 + \frac{M}{1024}\right) & \text{if } 1 \leq E \leq 30, \\ (-1)^S \infty & \text{if } \big(E = 31 \wedge M = 0\big), \\ \text{NaN} & \text{if } \big(E = 31 \wedge M > 0\big), \end{cases}$$

HDR Formats: comparisons

Encoding	Color Space	Bpp	Dynamic Range ($\log_{10}$)	Relative Error (%)
IEEE RGB	full RGB	96	79	0.000003
RGBE	positive RGB	32	76	1.0
LogLuv24	logY + (u,v)	24	4.8	1.1
LogLuv32	logY + (u,v)	32	38	0.3
Half RGB	RGB	48	10.7	0.1

Questions?