SAND2014-17776PE

Realistically Bad Data

Liam Boone

September 29th, 2014

U.S. DEPARTMENT OF ENERGY
National Nuclear Security Administration

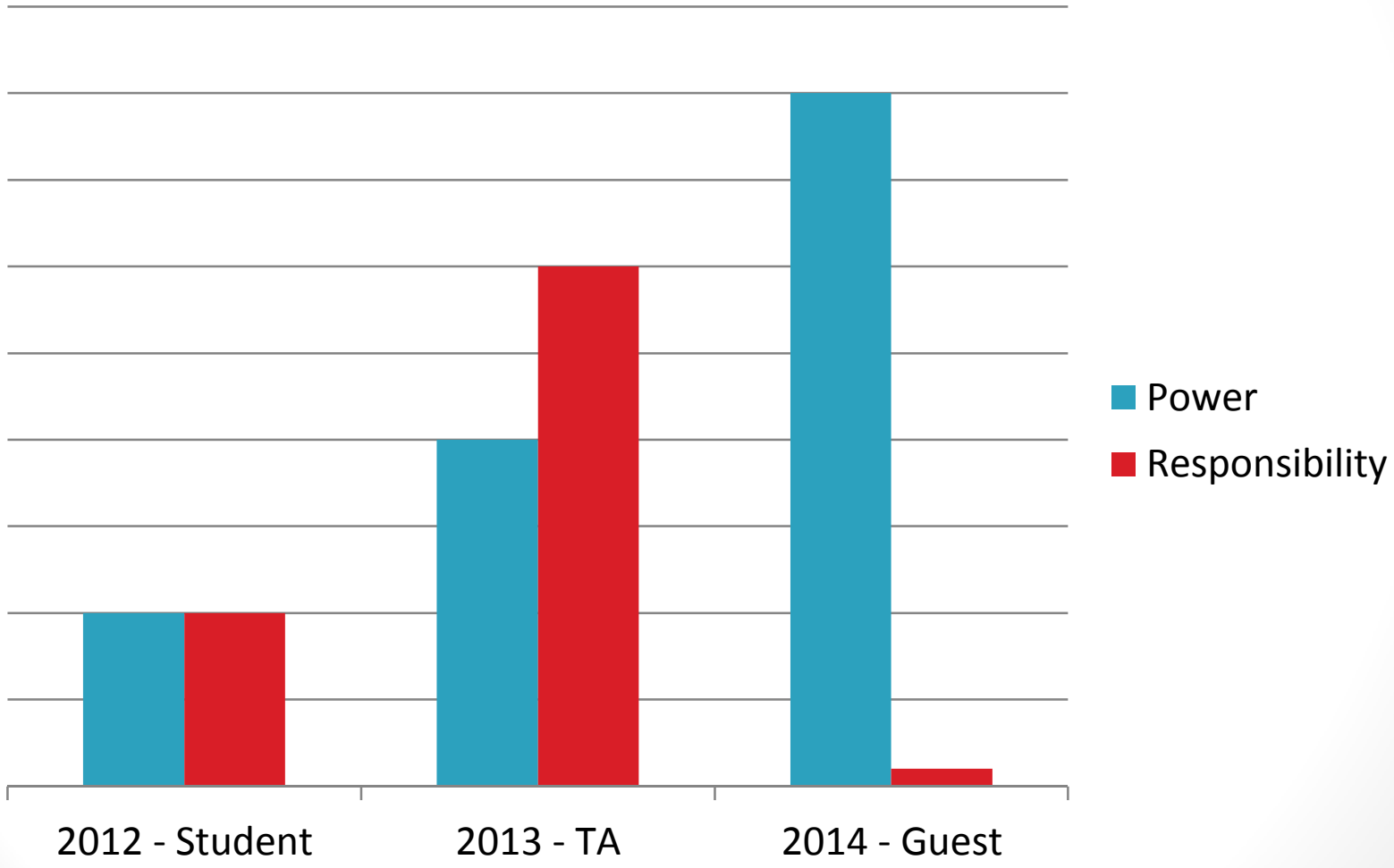
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

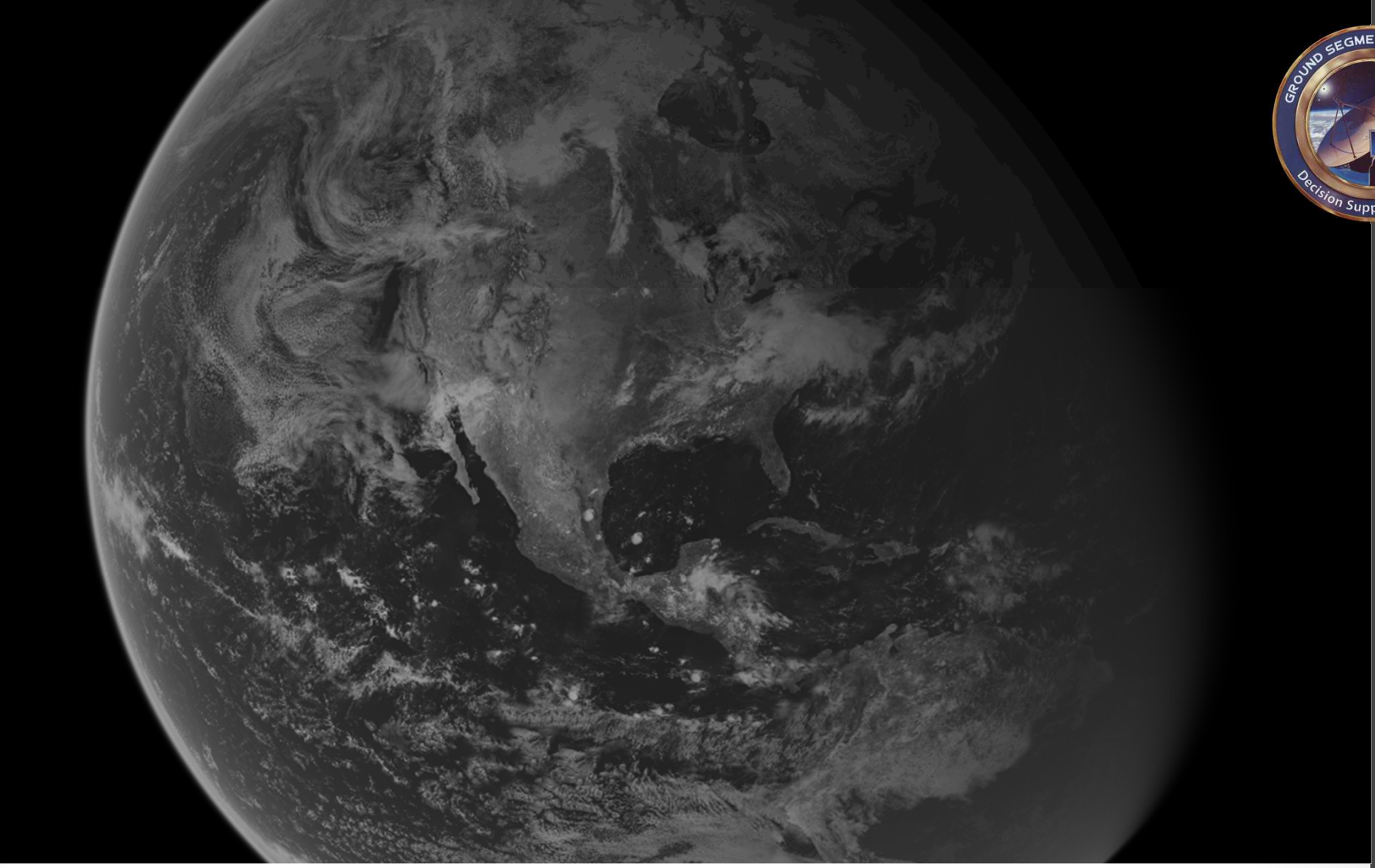
Sandia National Laboratories

About Me

- Pennsylvania State University (2007 – 2012)
  - B.S. in Electrical Engineering
  - B.S. in Computer Engineering
- University of Pennsylvania (2012 – 2014)
  - M.S.E. in Computer & Information Science
- Sandia National Labs (2012 – Now)

My History with CIS 565





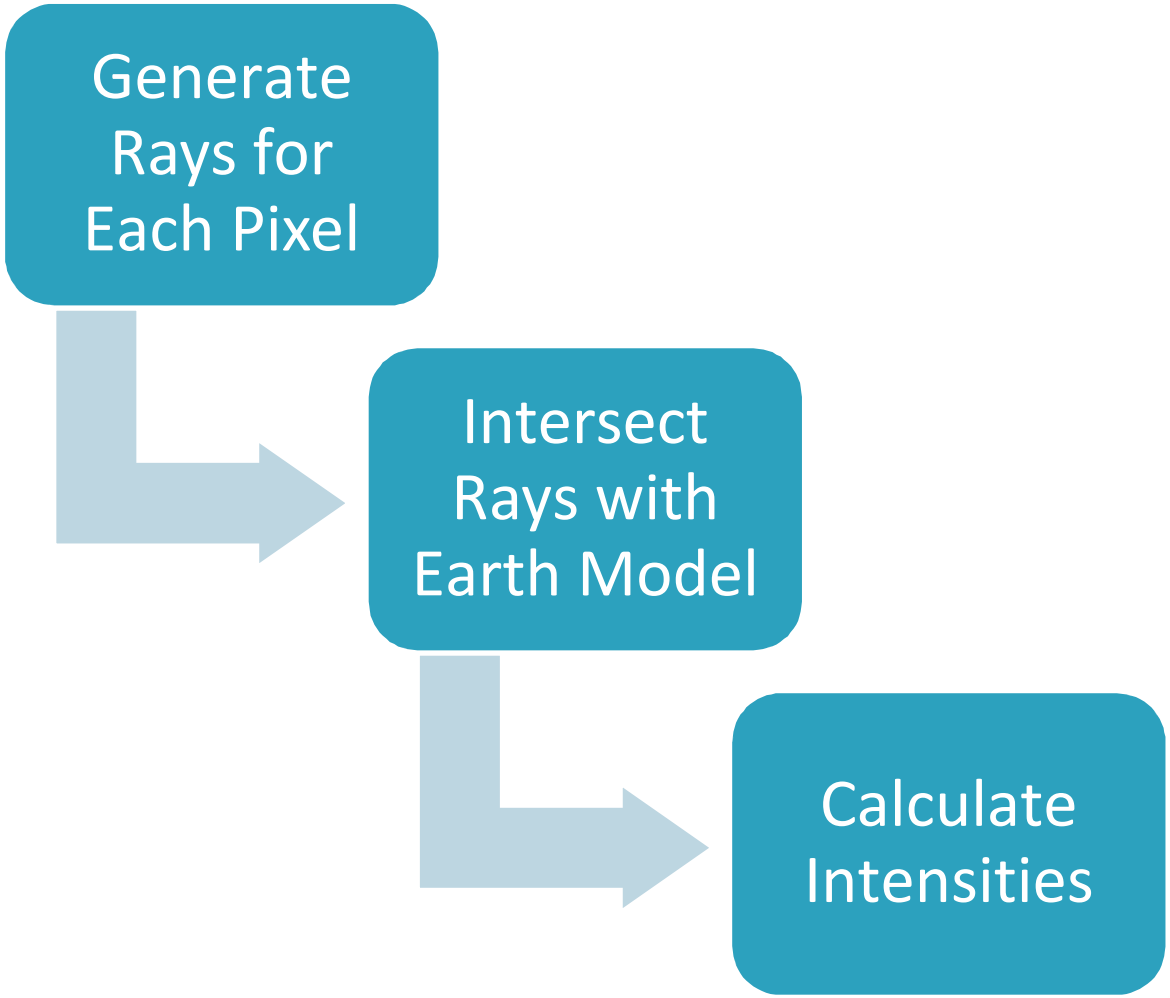
Globe Rendering

All world and cloud textures courtesy of NASA's MODIS datasets

Motivation

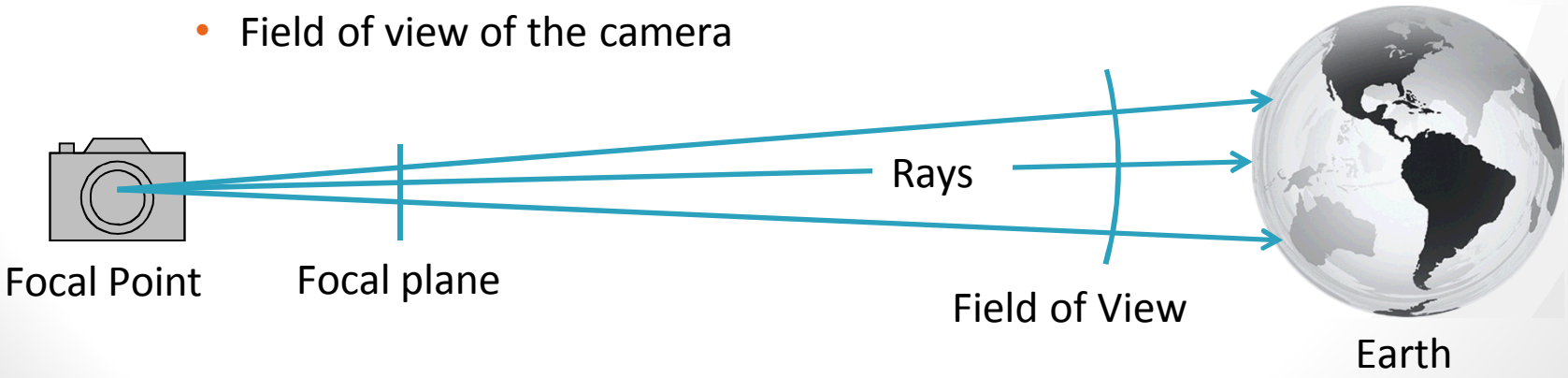
- Strong need for payload agnostic simulation of real-time data
- Introduce more dynamic scene elements to allow greater flexibility for algorithm developers
  - Current test data has static backgrounds and little to no bad pixels
  - Would allow developers to test on realistic scenes
- Goals:
  - Faster development cycles and fewer bugs in software
  - Dynamically produce realistic scenes to provide developers with a larger library of test scenarios

Processing Flow



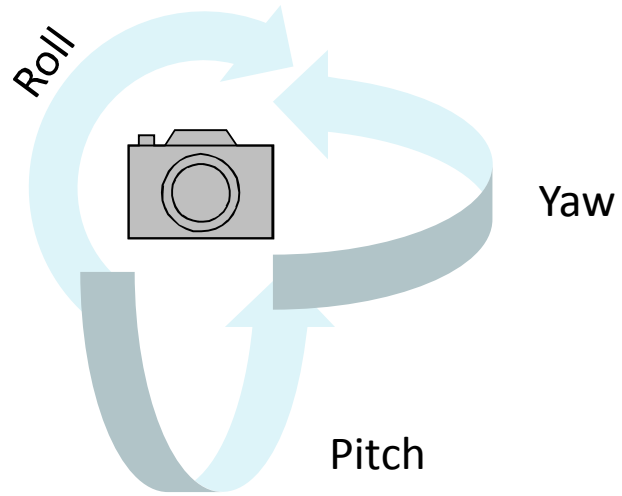
Generate Rays

- Rays form a one-to-one correspondence with Pixels in the simulated focal plane
  - Each Ray originates at the center of an idealized camera located at some position in space and extends through an imaginary focal plane located some distance in front of the idealized camera
  - For this we need to know:
    - Position of the camera
    - Rotation of the camera (to determine where it is looking)
    - Resolution of the focal plane (number of pixels wide/tall)
    - Field of view of the camera



Sensor Jitter

- The camera is not perfect and will move from frame to frame
- Represented as a rotation from true pointing direction
  - Euler angles (Roll, Pitch, Yaw)
  - Very small
  - Random values sampled from N-D Gaussian distribution
  - Over time the look angle will walk off from where it should be



Ray Intersection

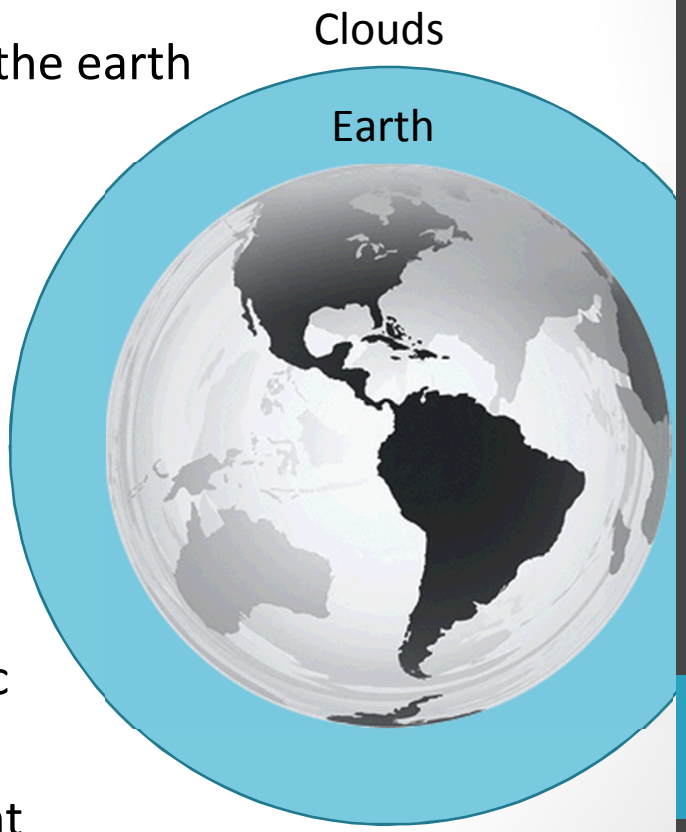
- All models use simple spheres as primitives
  - Straight-forward intersection routine
  - Use formula for sphere ($x^2 + y^2 + z^2 = r^2$)

    and ray ($\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + t \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix}$) to derive an intersection point
  - Boils down to a familiar quadratic equation ($At^2 + Bt + C = 0$)

- Helps with computational complexity
  - Code reuse
  - Easy to manage

Earth Model (Simple)

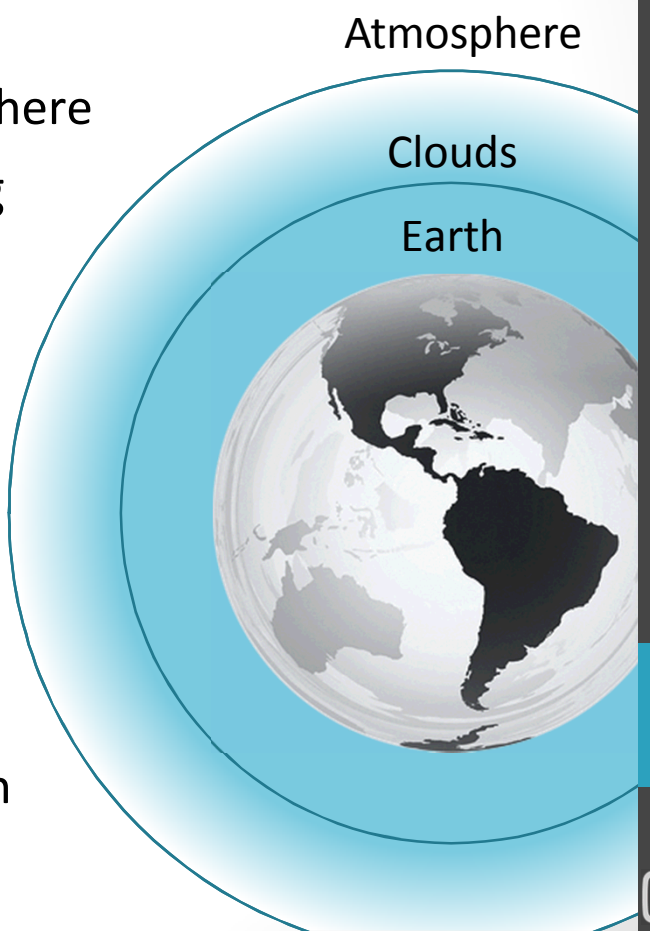
- Simple Earth Model
  - Two concentric spheres
  - Inner sphere represents the surface of the earth
  - Outer sphere is the cloud layer
- Details are represented by textures
  - No height data involved
  - Intensity values from NASA MODIS
  - Clouds are “flattened” into one layer



- Pros
  - Easy to code
  - Fast
- Cons
  - Not very realistic
  - Some test cases hard to represent

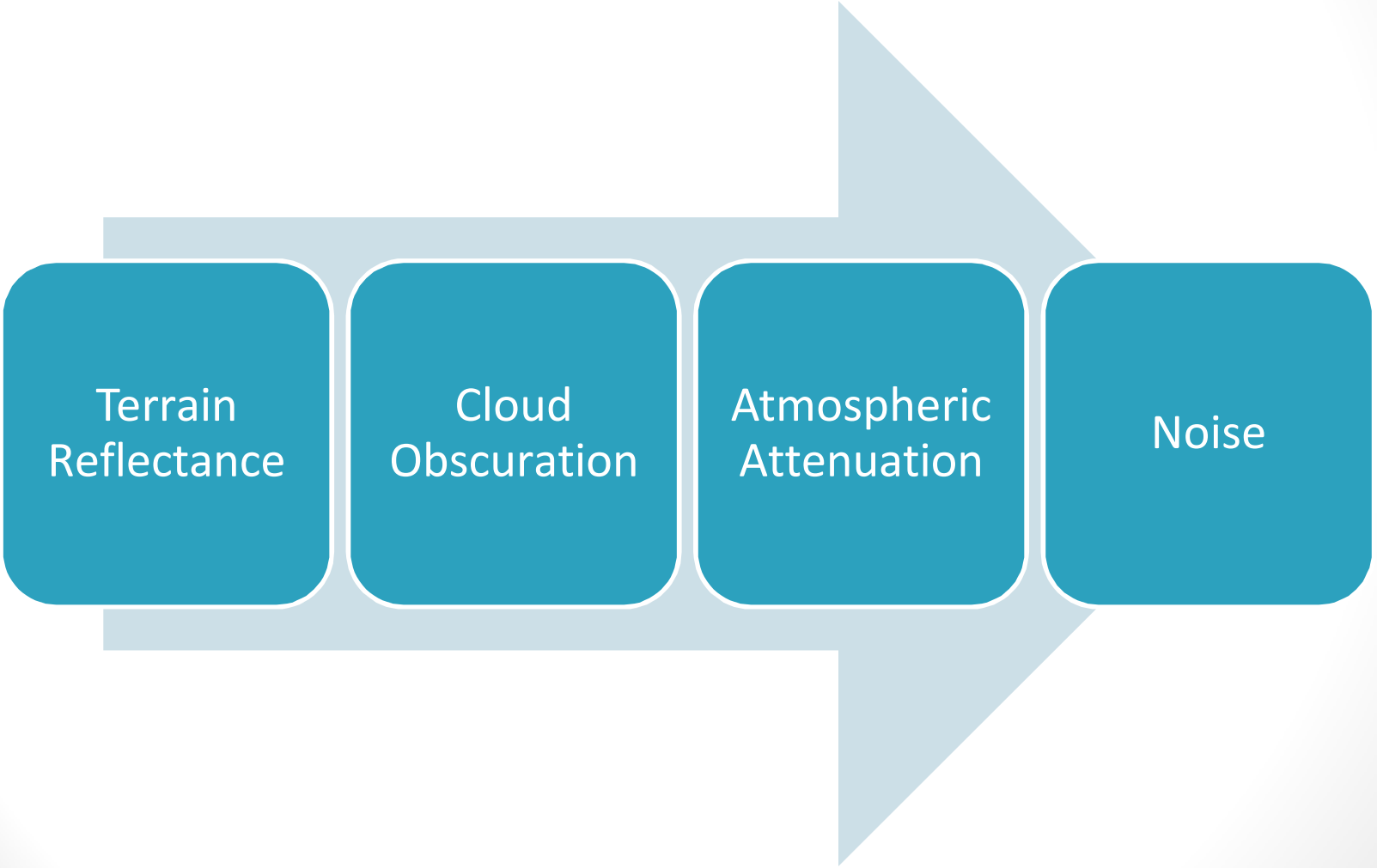
Earth Model (Complex)

- Complex Earth Model
  - Three concentric spheres
  - Inner two spheres as before
  - Outer sphere is the extent of the atmosphere
- Atmosphere layer contributes scattering
  - Rayleigh and Mie scattering
  - Must evaluate a double integral per pixel
  - Wavelength dependent



- Pros
  - More accuracy
  - More test cases
- Cons
  - Still not perfect
  - Requires much more computation

Intensity calculation

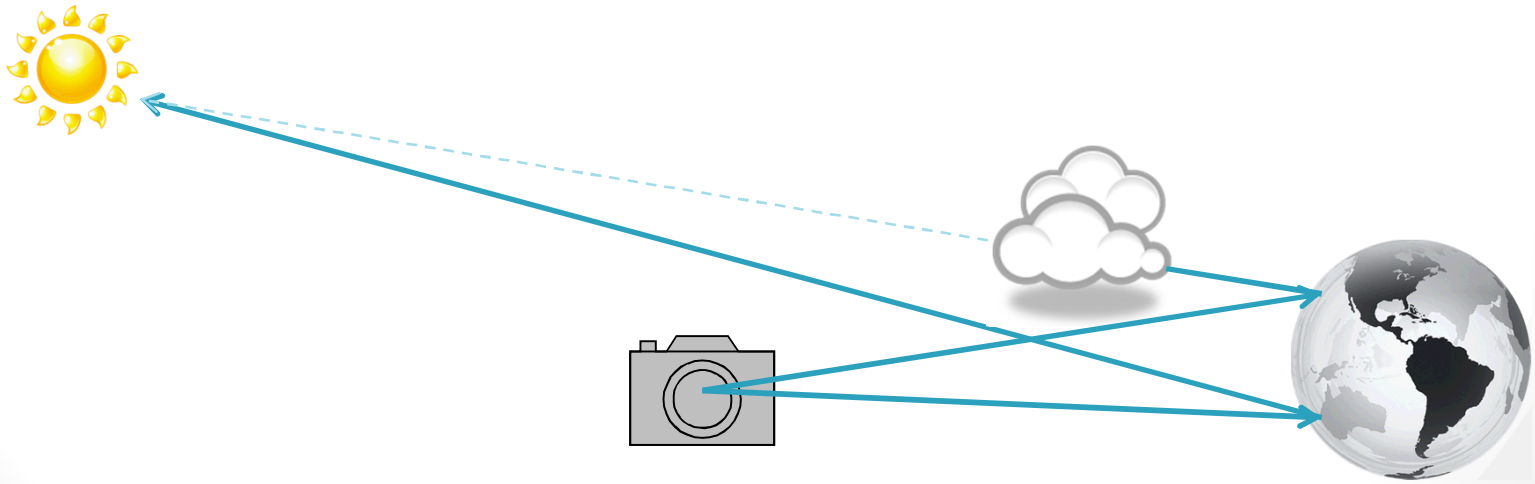


Terrain

- From the ray/sphere intersection point determine a Lat/Lon coordinate
- Sample from an image representing the surface of the earth to get the base intensity
- Sample from a binary image representing whether a given coordinate is water or not to get the specular coefficient
- Use simple Phong lighting model to apply diffuse and specular components of the sun's light

Clouds

- From the ray/sphere intersection point determine a lookup coordinate
- Sample the cloud image to determine a cloud density value
- Use the density value to attenuate the intensity of the ground
- Can also use a second ray cast to the sun to determine if the clouds will cast a shadow on the surface of the earth



Atmosphere I

- Attenuation based on distance light travels through the atmosphere
- Calculated with a double integral

Attenuation Factor:

$$K(\lambda) = \frac{K_0}{\lambda^4}$$

Phase Function:

$$F(\theta, g) = \frac{3(1-g^2)(1+cos^2\theta)}{2(2+g^2)(1+g^2+2g\,cos\theta)^{\frac{3}{2}}}$$

Out-Scattering Function:

$$t(P_a, P_b, \lambda) = 4\pi K(\lambda) \int_{P_a}^{P_b} e^{-\frac{h}{H_0}}\, ds$$

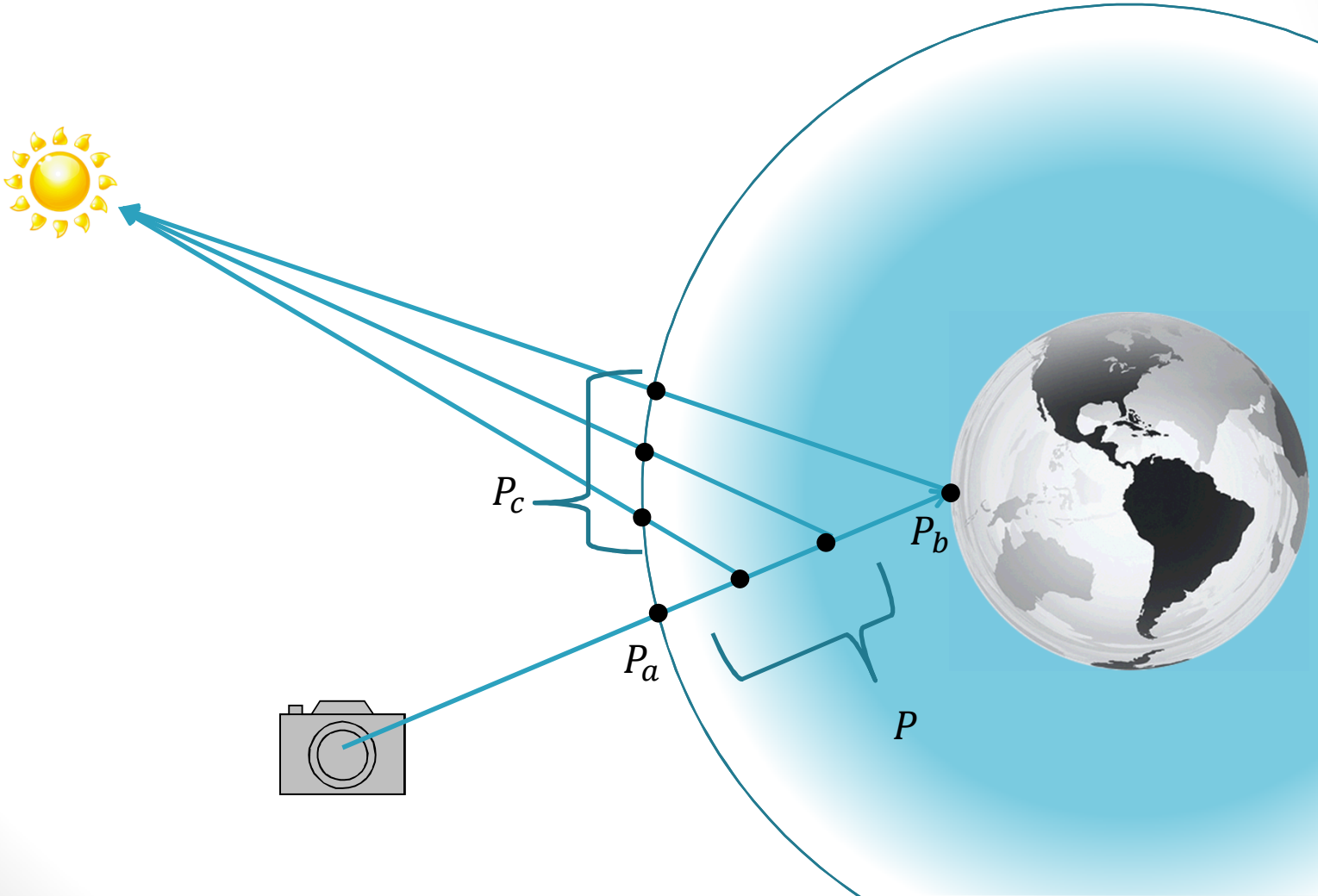
In-Scattering Function:

$$I_v(\lambda) = I_s(\lambda)\mathrm{K}(\lambda)F(\theta, g) \int_{P_a}^{P_b} e^{-\frac{h\left(-t(P,P_c,\lambda)-t(P,P_a,\lambda)\right)}{H_0}} ds$$

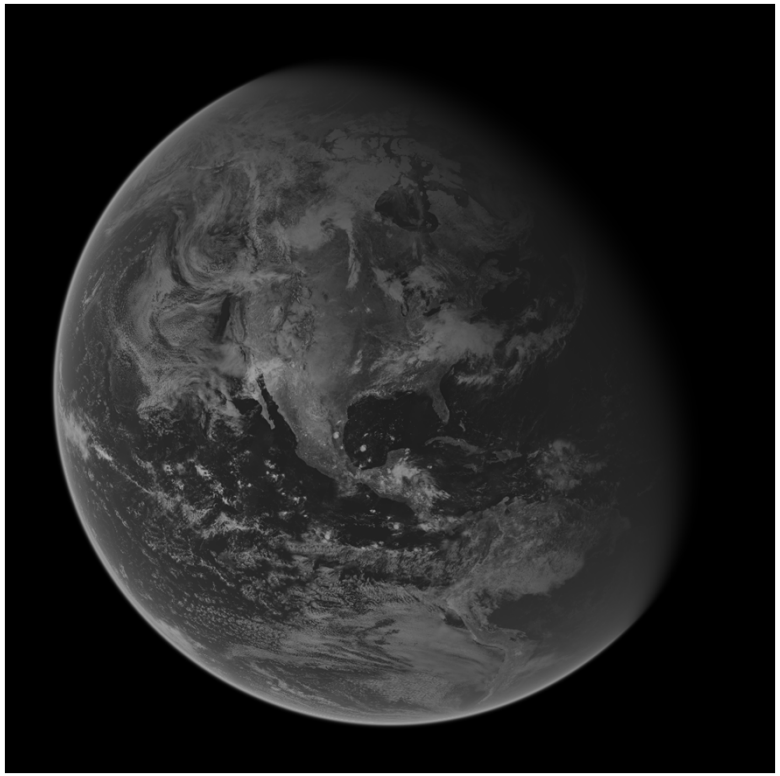
Atmosphere II

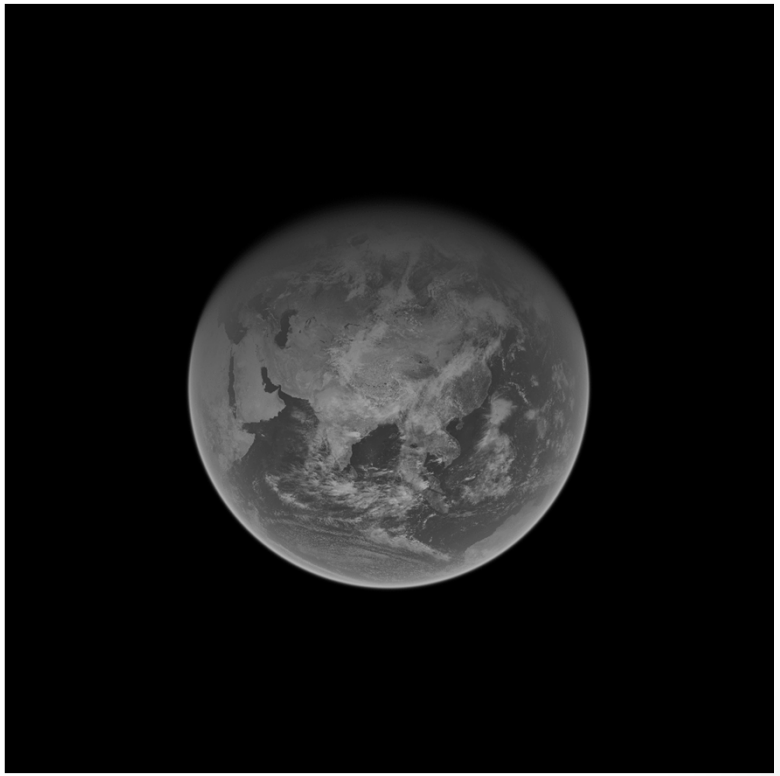
- K_0: Molecular density at sea level
- λ: Wavelength of detectable light
- θ: Angle between incident light and viewing vector
- g: Scattering parameter
  - 0 for Rayleigh
  - Between -0.75 and -0.999 for Mie
- H_0: Scale height of the atmosphere
- h: Height above surface of the earth
- P, P_x: Points (see figure on next slide)

Atmosphere III



Scattering Results





All world and cloud textures courtesy of NASA's MODIS datasets

Intensity Calculation (Noise)

- Shot noise
  - Simulates a more grainy image
  - Use a Poisson distribution centered on the pixel value
  - Variable with time
- Popcorn noise
  - Simulates under/over-excited pixels
  - Sample a uniform distribution and saturate or zero the pixel if the sample is above or below some threshold respectively
  - Variable with time
- Fixed pattern noise
  - Simplest noise type
  - Defines a fixed pattern of dead/under-excitable pixels
  - Constant over the lifetime of the simulation

Shot Noise

- Knuth's Poisson sampling algorithm:

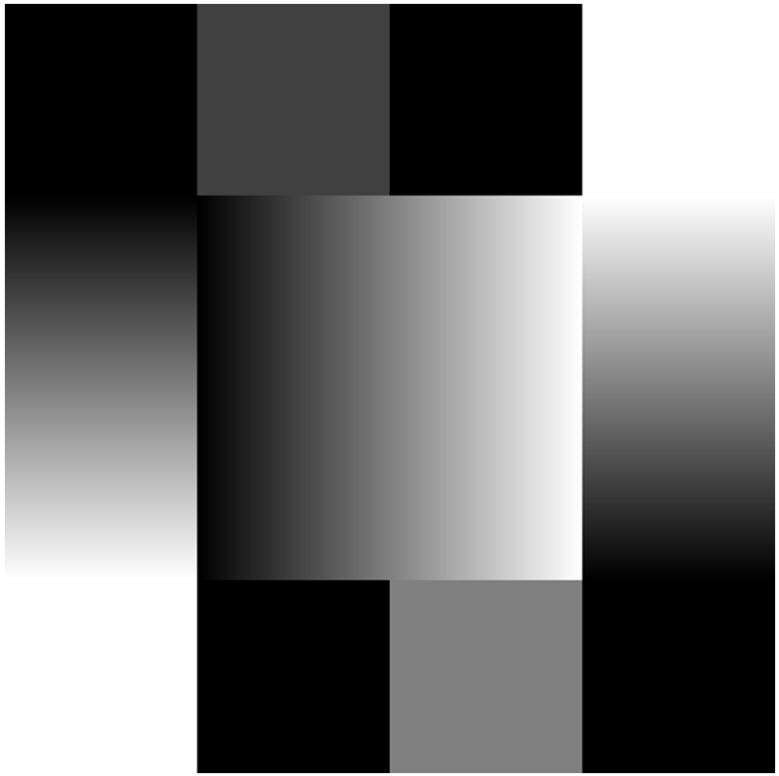
```
Def Poisson(λ):
    Let L ← exp(−λ), k ← 0 and p ← 1.
    do:
        k ← k + 1.
        Generate uniform random number u in [0,1].
        let p ← p × u.
    while p > L.
    return k − 1.
```

- Can be parameterized by multiplying λ by some constant s before sampling and then dividing by the same constant afterwards
  - Small values of s lead to more grainy images
  - Large values of s cause the algorithm to perform slower

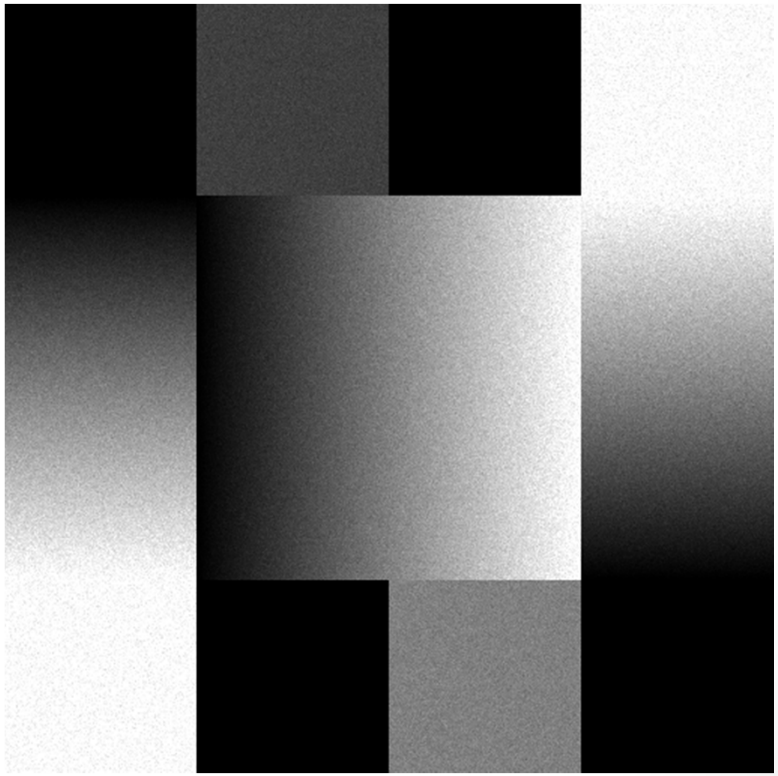
```
Poisson(λ*s)/s
```

Shot Noise

Test Pattern

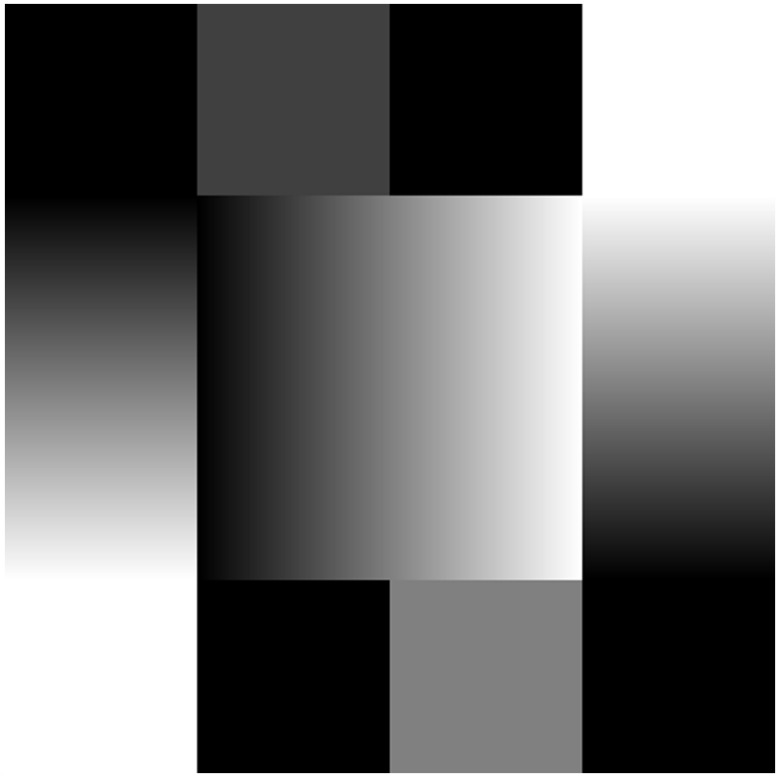


Noise (s=2)

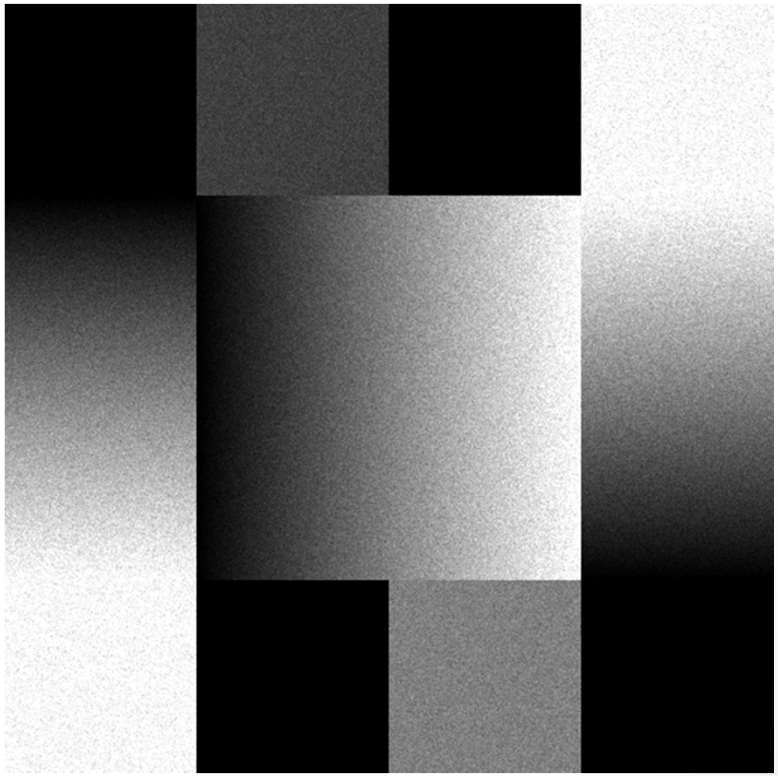


Shot Noise

Test Pattern

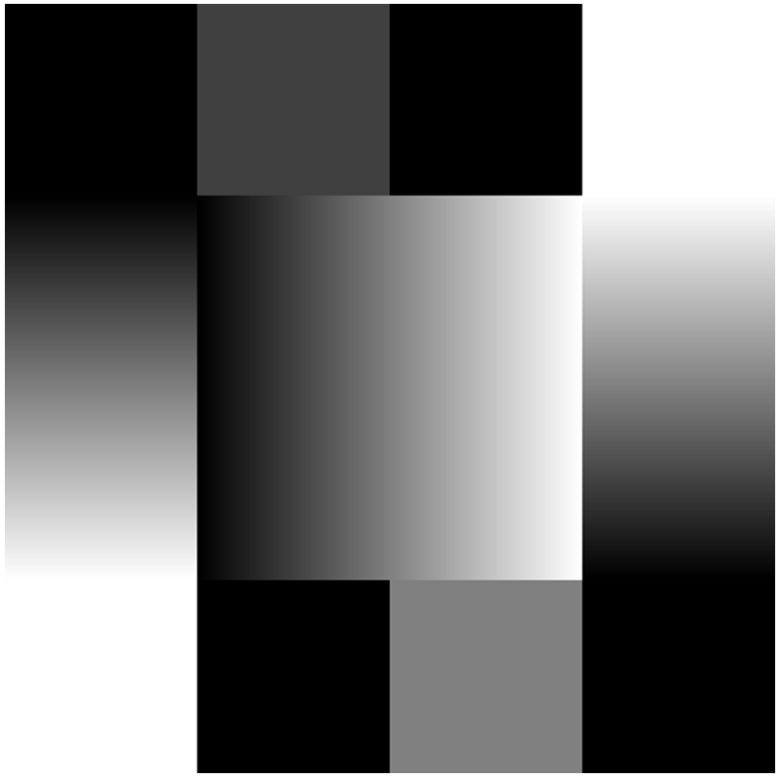


Noise (s=1)

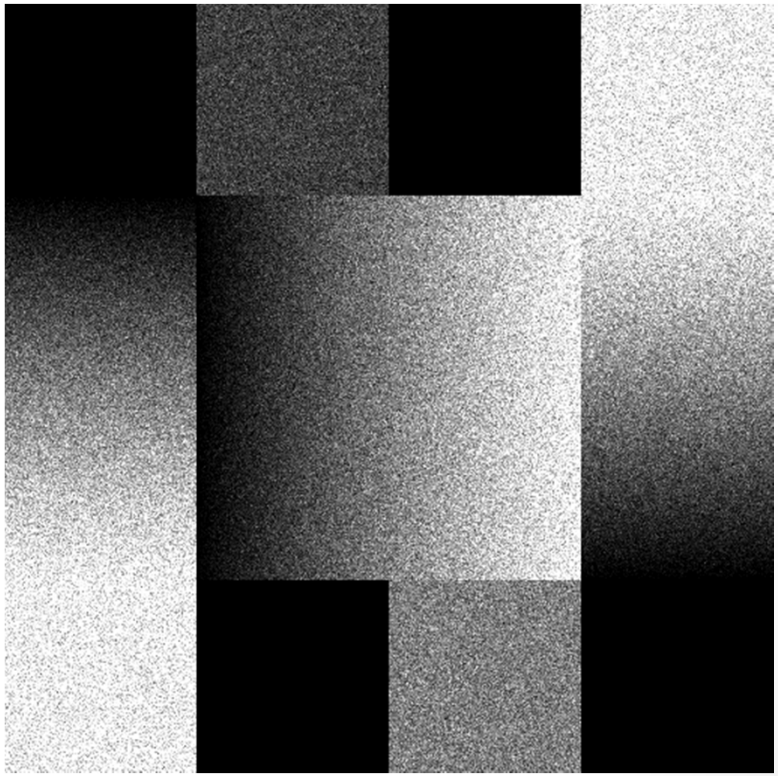


Shot Noise

Test Pattern



Noise (s=0.1)



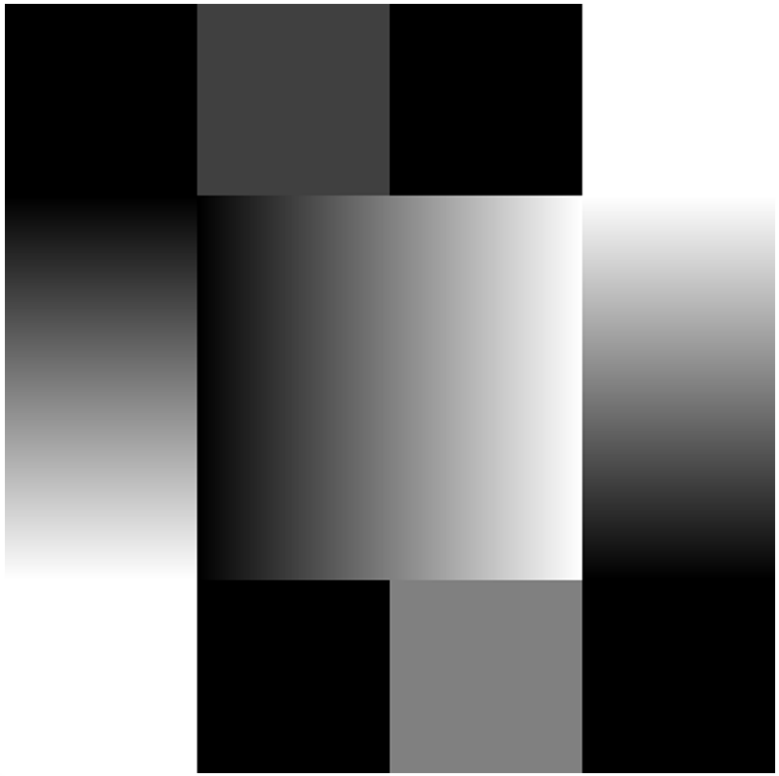
Popcorn Noise

- Popcorn sampling algorithm:

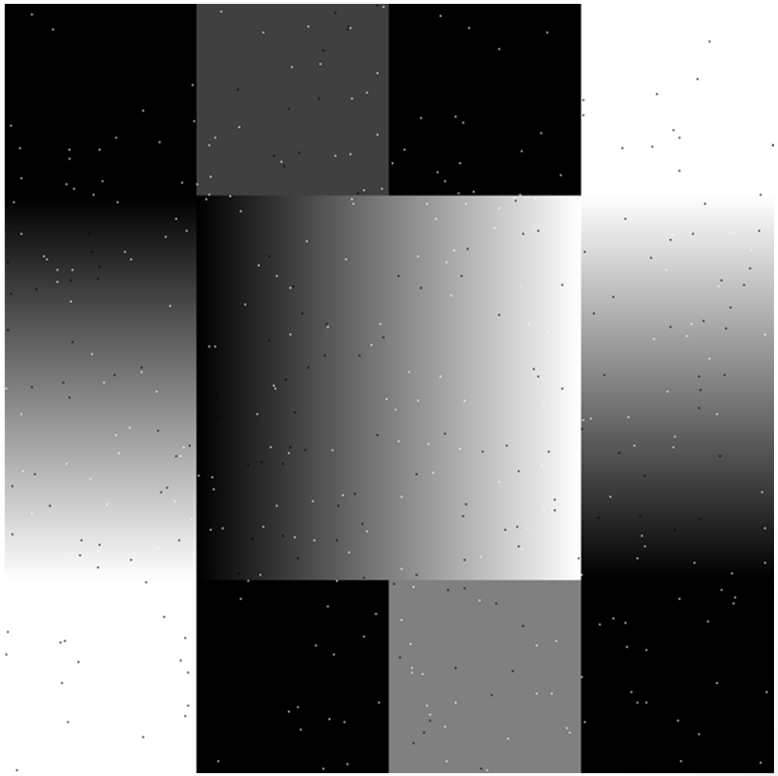
```
Def Popcorn(x, t):
    Generate uniform random number u in [0,1].
    If u > (1-t):
        return 255.
    If u < t:
        return 0.
    return x.
```

Popcorn Noise

Test Pattern

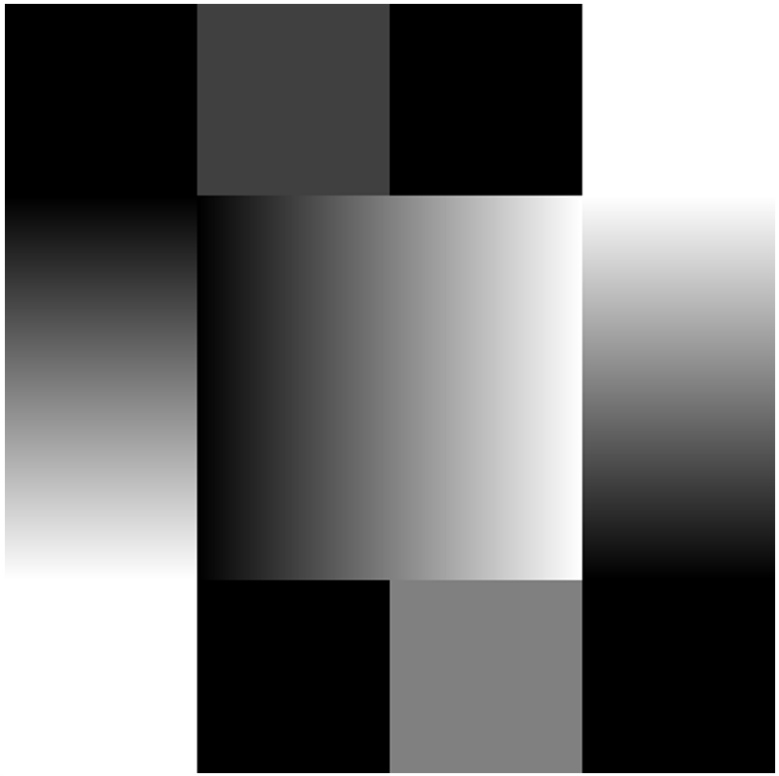


Noise (t=0.001)

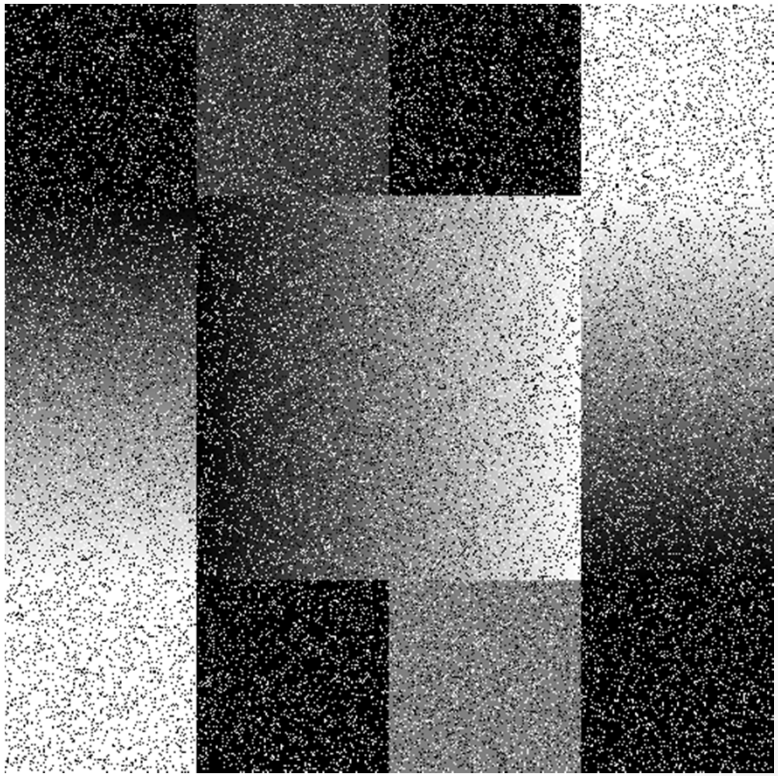


Popcorn Noise

Test Pattern



Noise (t=0.1)

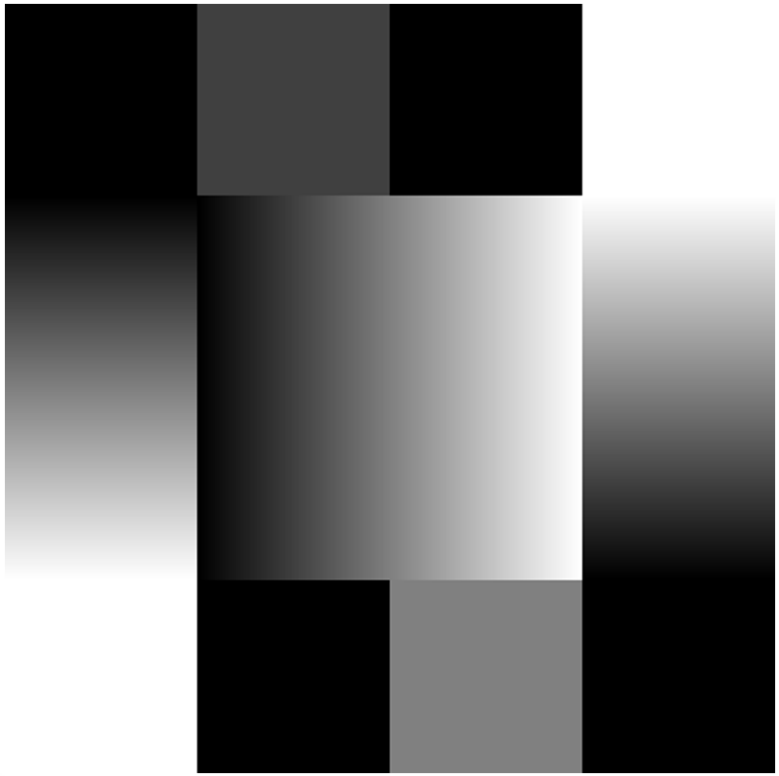


Fixed Pattern Noise

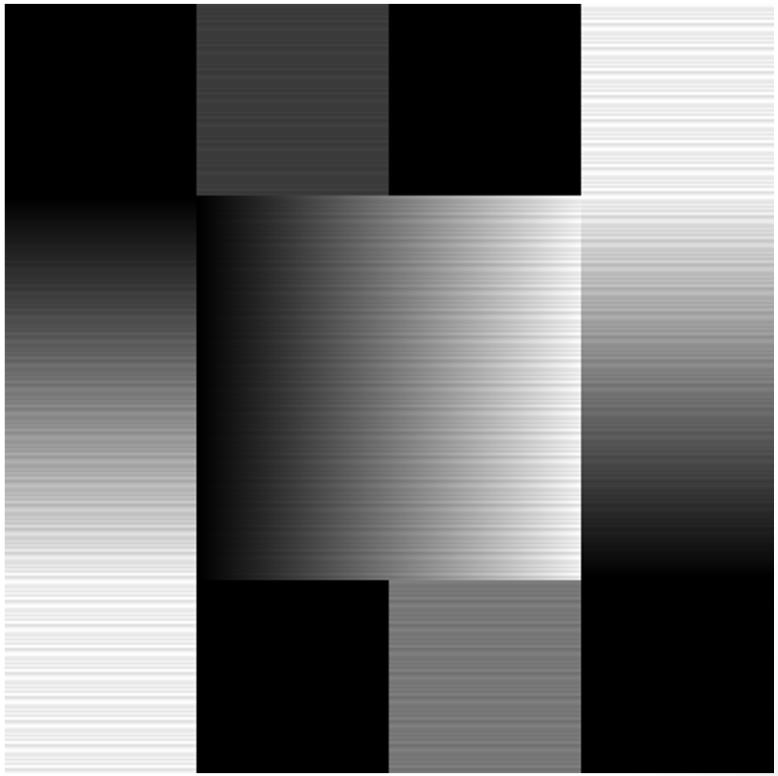
- Fixed pattern noise is simply a constant offset or attenuation applied to the whole FPA
- Simulates manufacturing defects and errors present in the hardware

Fixed Pattern Noise

Test Pattern

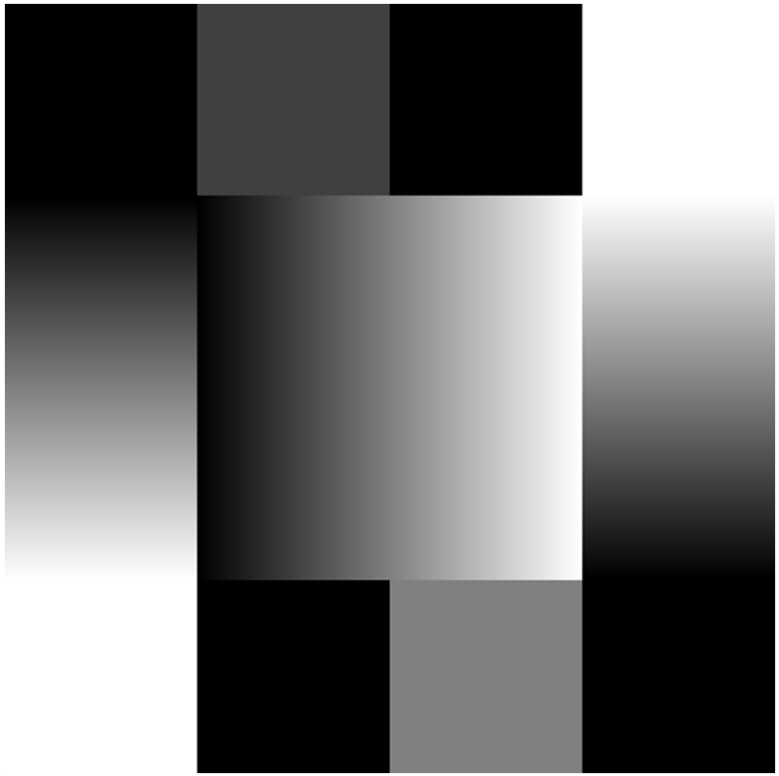


Noise (Horizontal Bars)

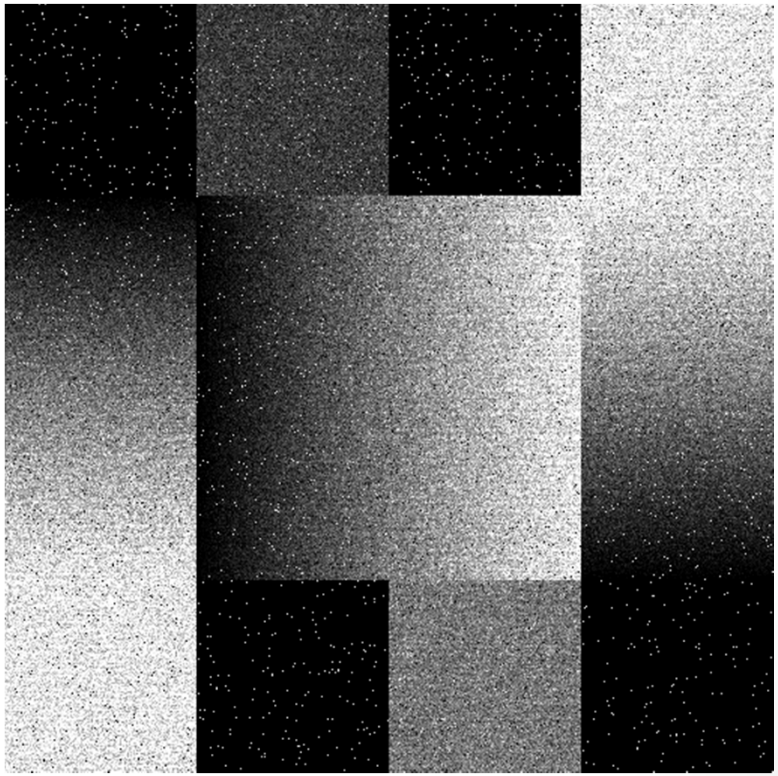


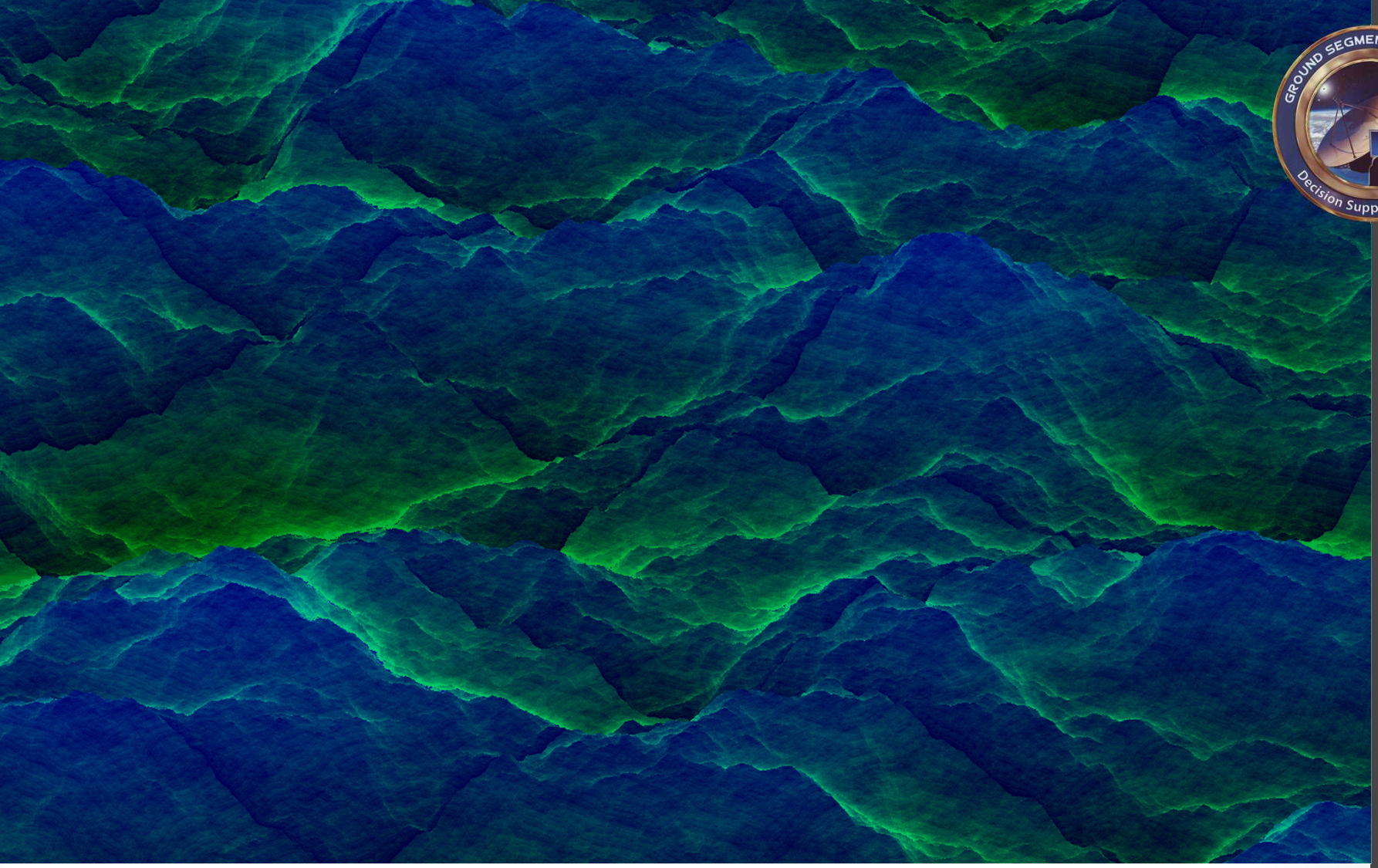
All Together

Test Pattern



Noise (p=0.1, t = 0.01, Bars)





Terrain Rendering

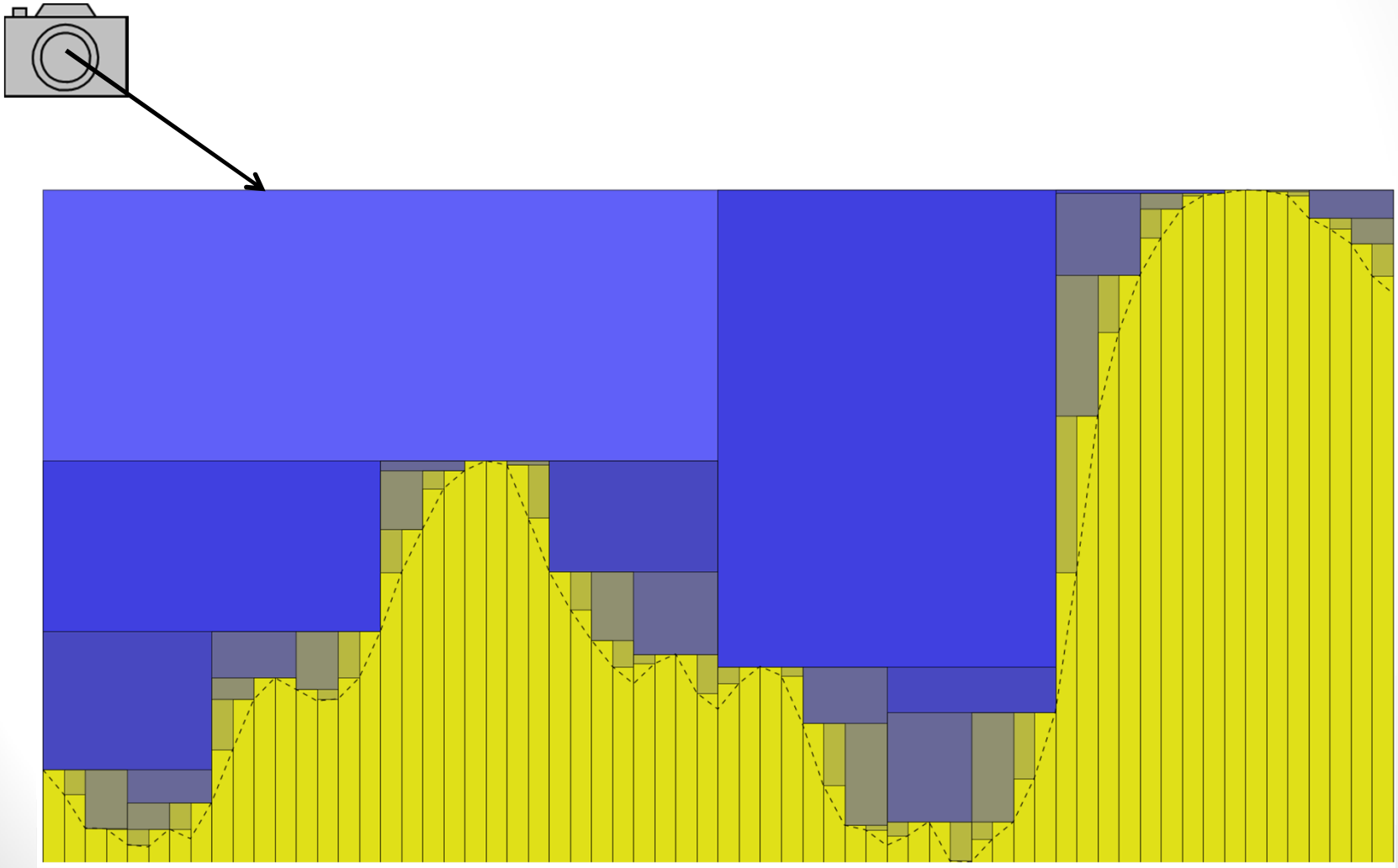
Overview

- Atmospherics and sensor noise are great first steps
- Good enough for some situations
- Sometimes necessary to fully model the terrain
  - Need terrain data to answer questions about where mountain shadows lie
  - Is a particular asset's view obstructed by terrain in any way?
- Current dataset chosen is DTED level 1
  - Partial global coverage
  - 90m post spacing in most of the world

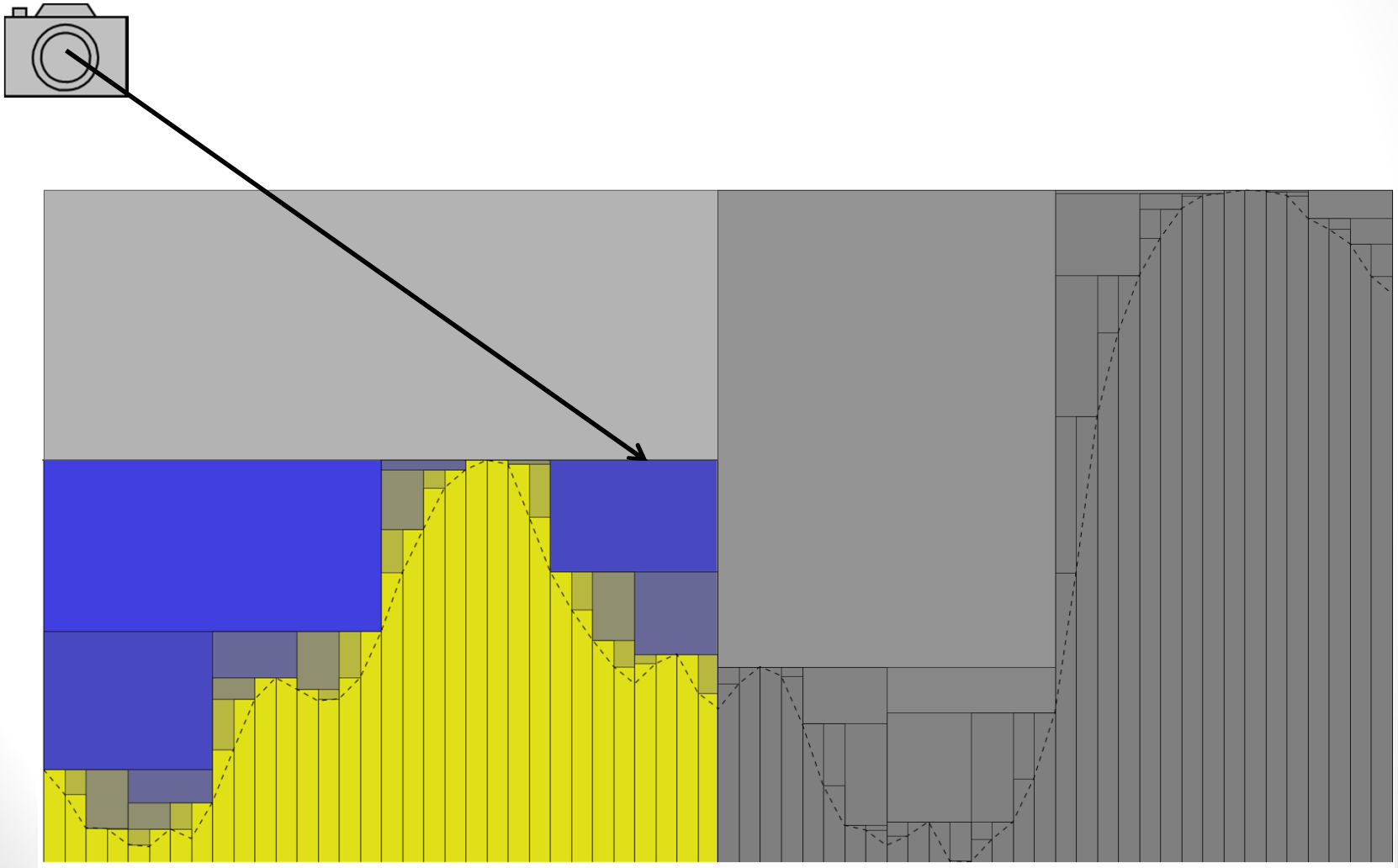
Algorithm

- Preprocess data into a chunked maximum mipmap format
  - Each entry in each level of the mipmap represents the maximum height of the four entries in the next finer level
  - A 513x513 chunk of elevation posts give a 512x512 initial level
  - Generate coarser levels until one value is reached
  - Chunk edges will share posts
- Ray march through the mipmap structure
  - Step through progressively finer levels of the mipmap until the finest layer is encountered or the ray steps off of the chunk
- Intersect the ray with a bilinear patch to determine the final elevation of the terrain

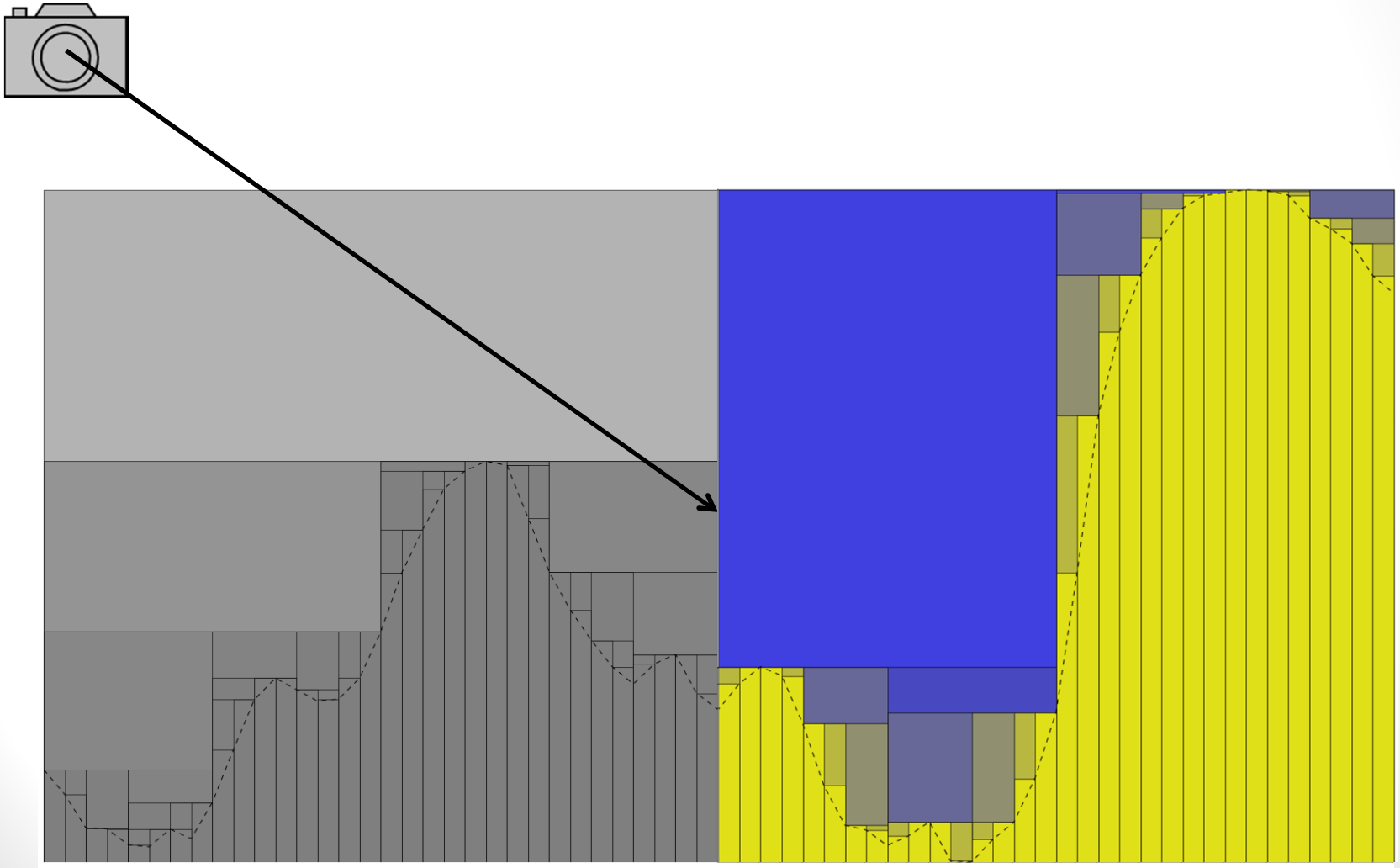
Maximum Mipmap (2D)



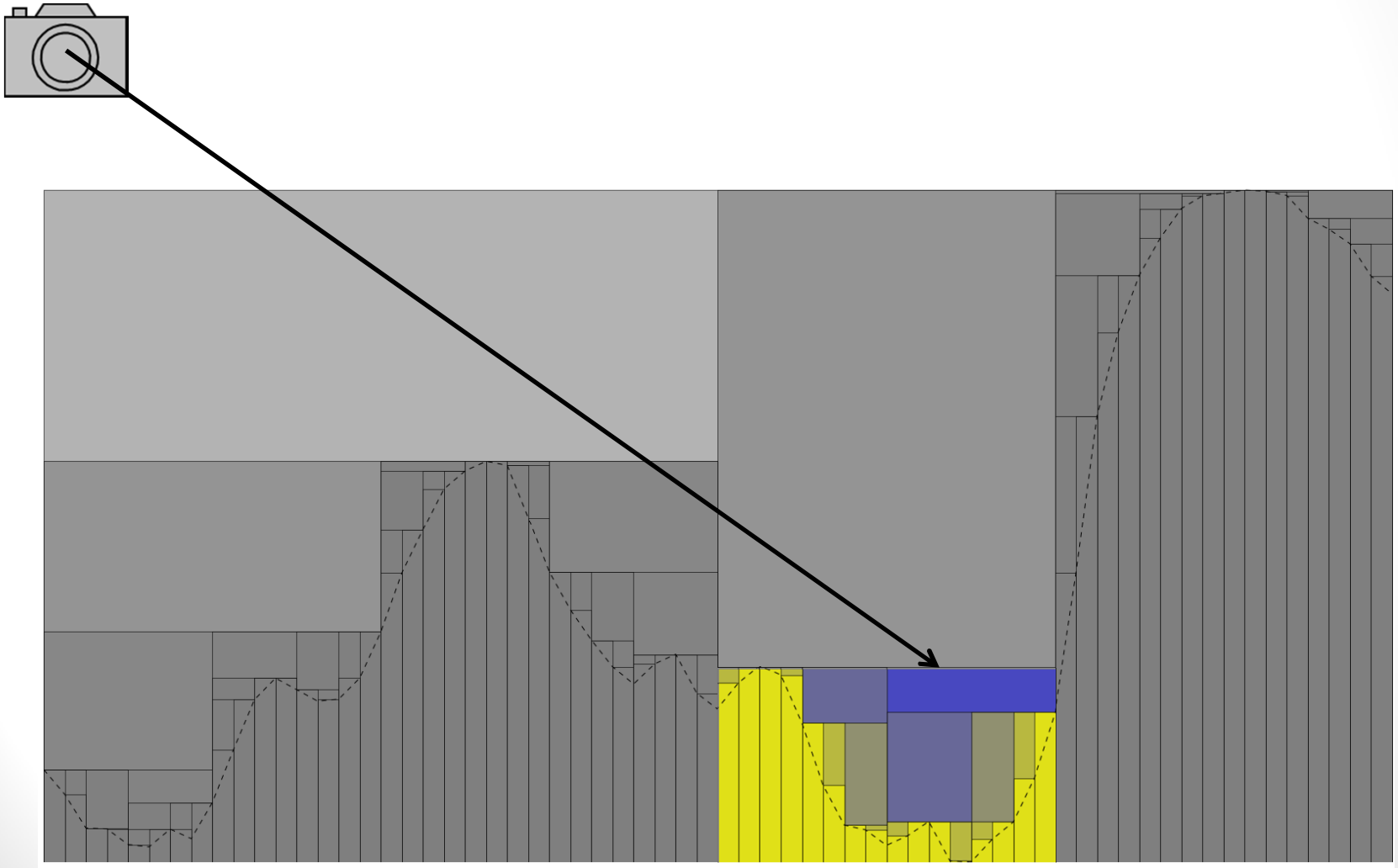
Maximum Mipmap (2D)



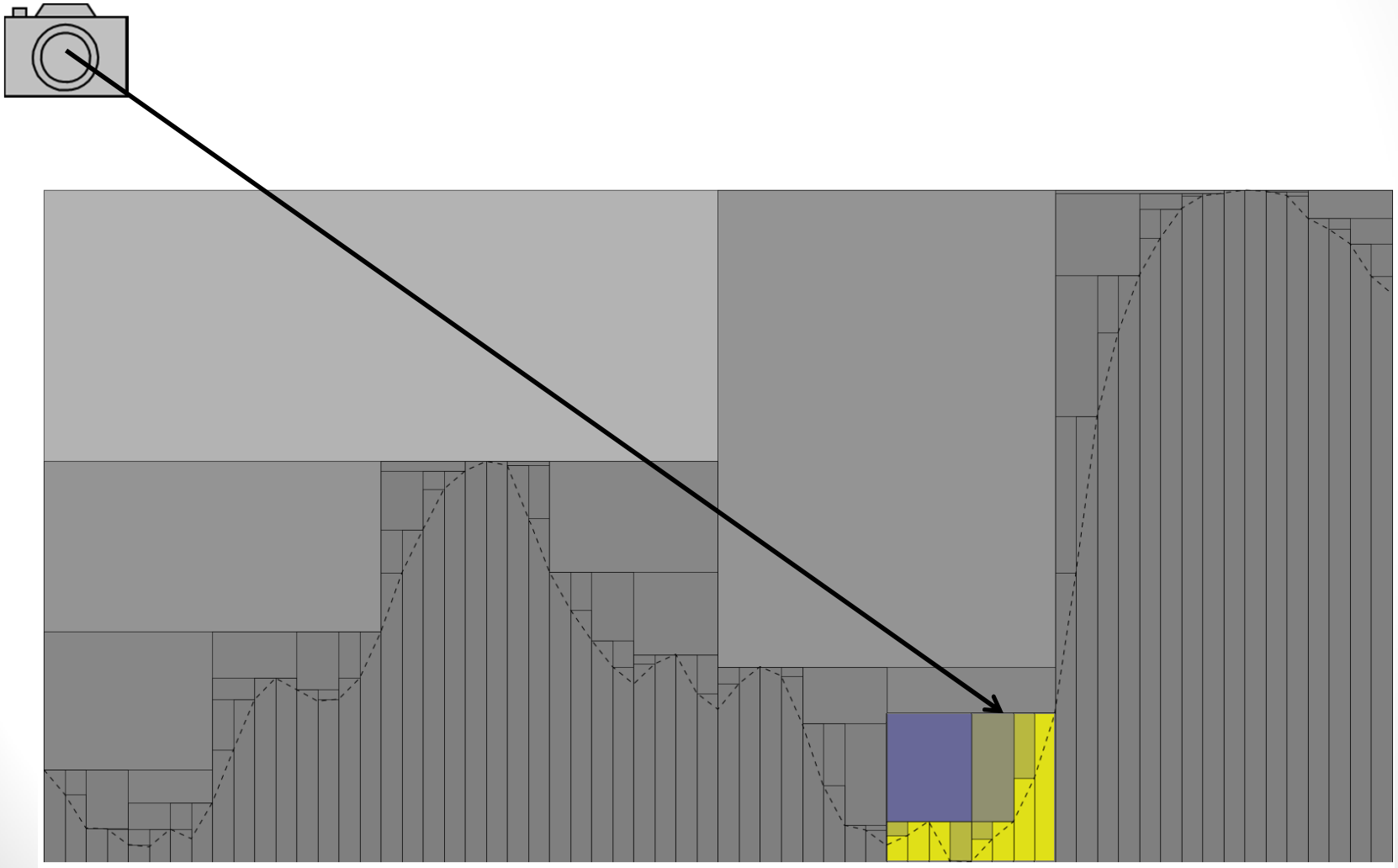
Maximum Mipmap (2D)



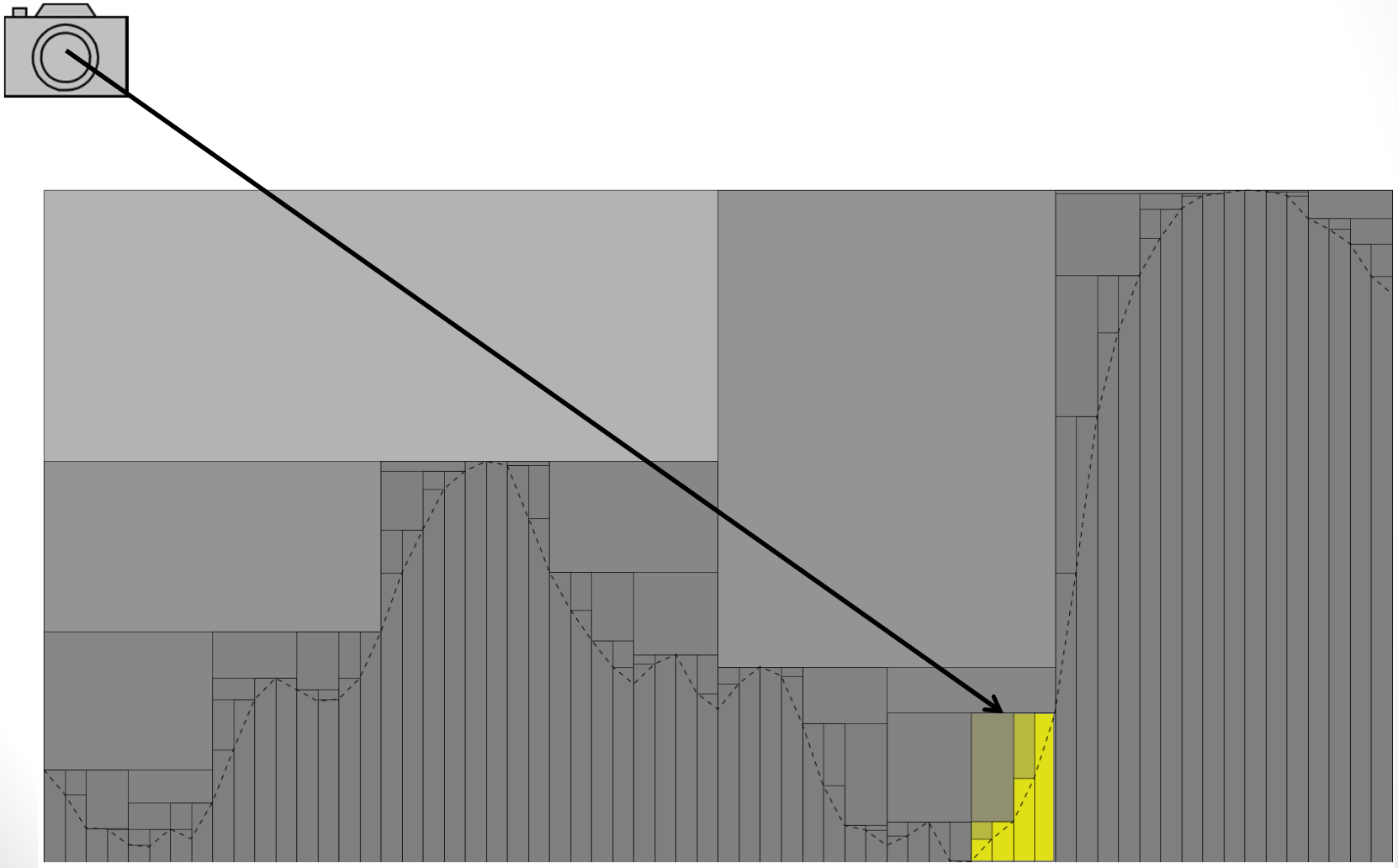
Maximum Mipmap (2D)



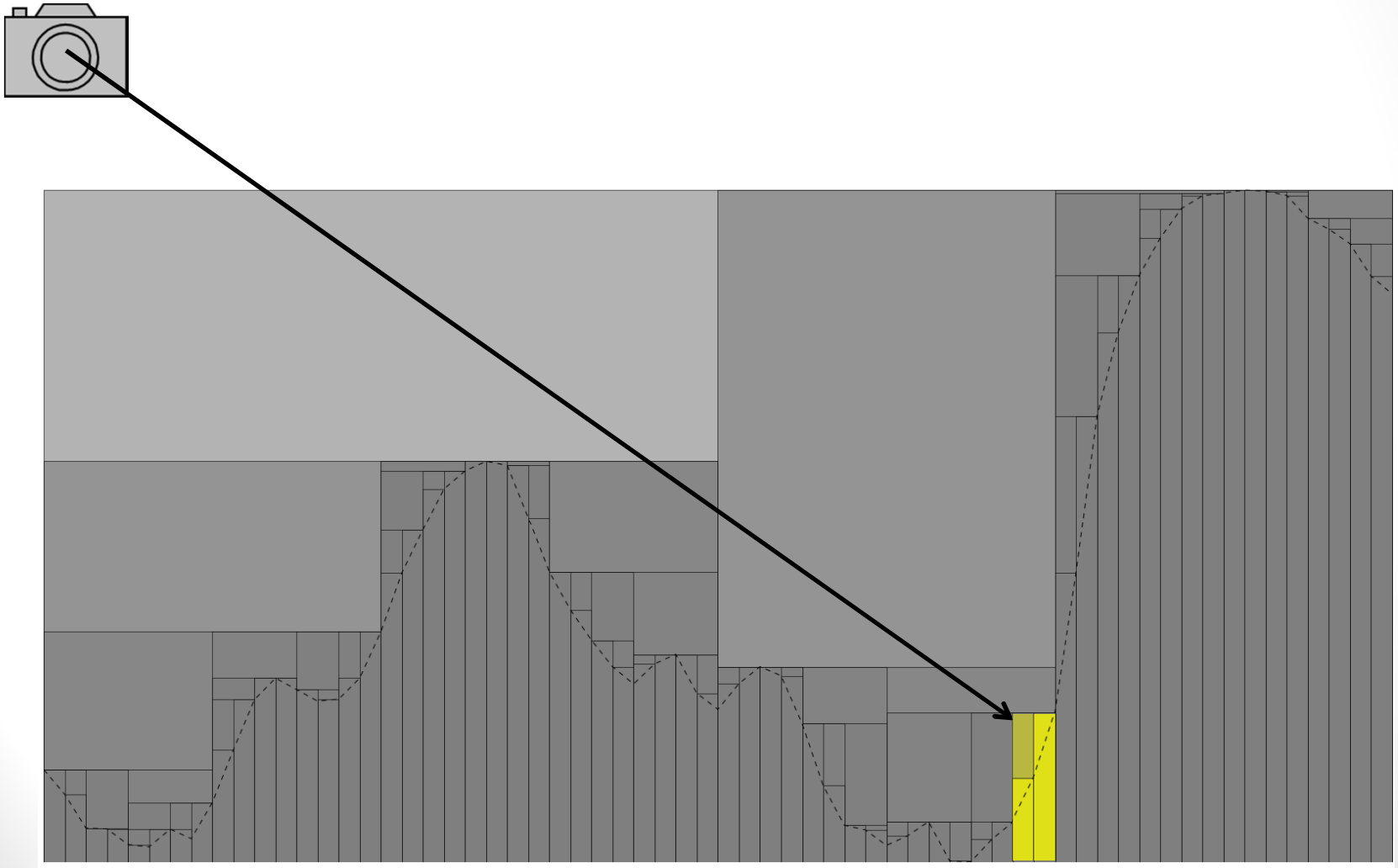
Maximum Mipmap (2D)



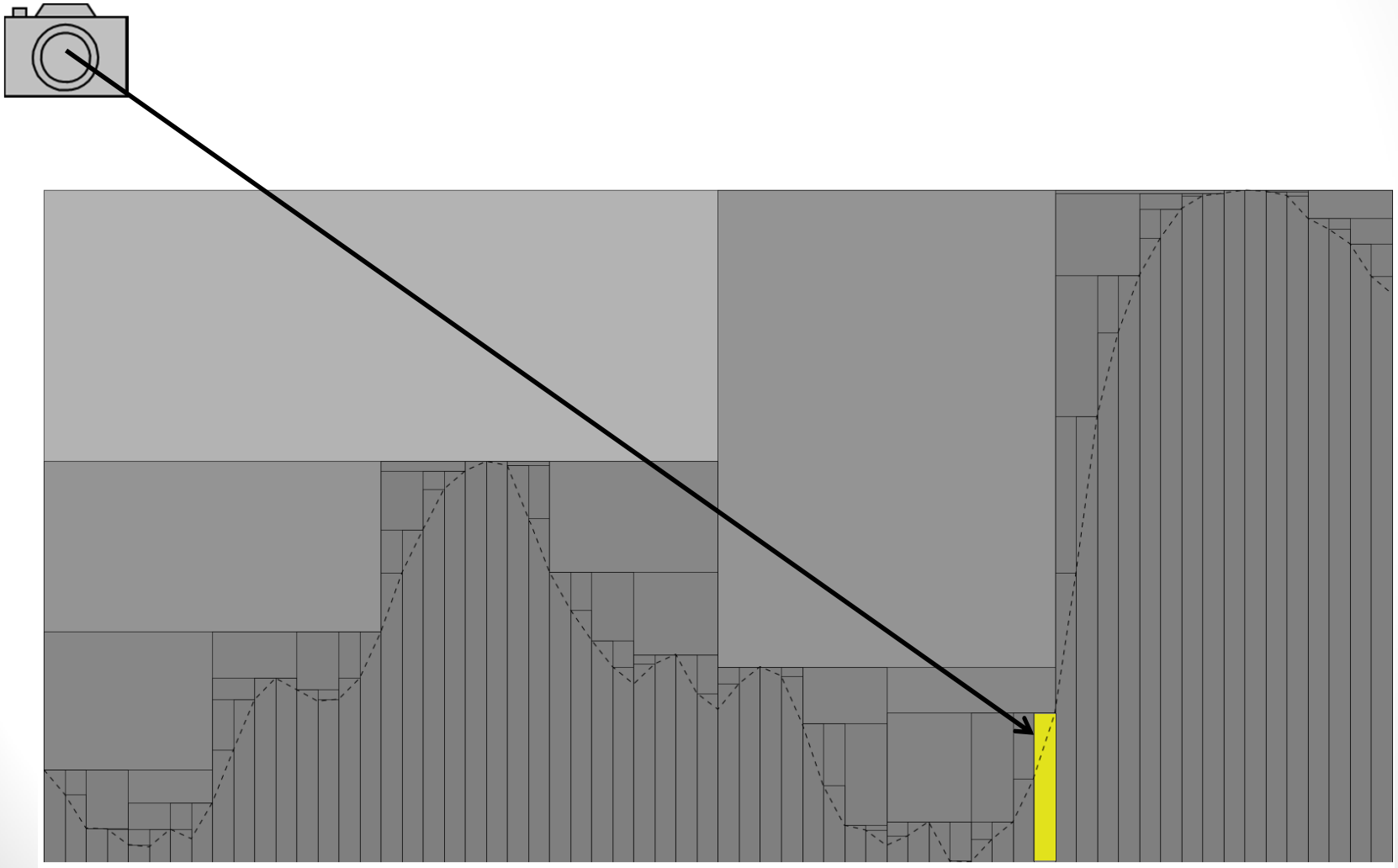
Maximum Mipmap (2D)



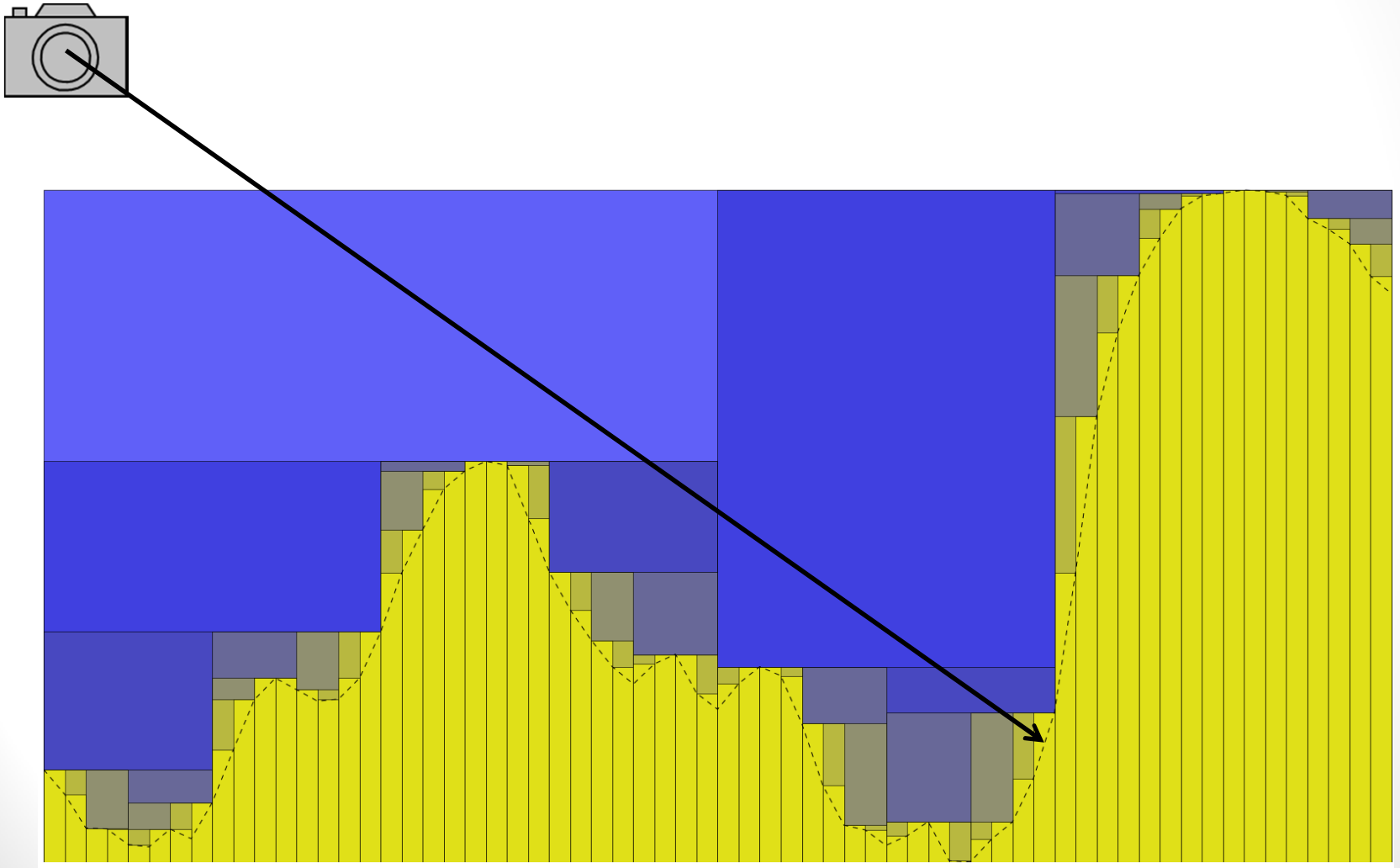
Maximum Mipmap (2D)



Maximum Mipmap (2D)



Maximum Mipmap (2D)

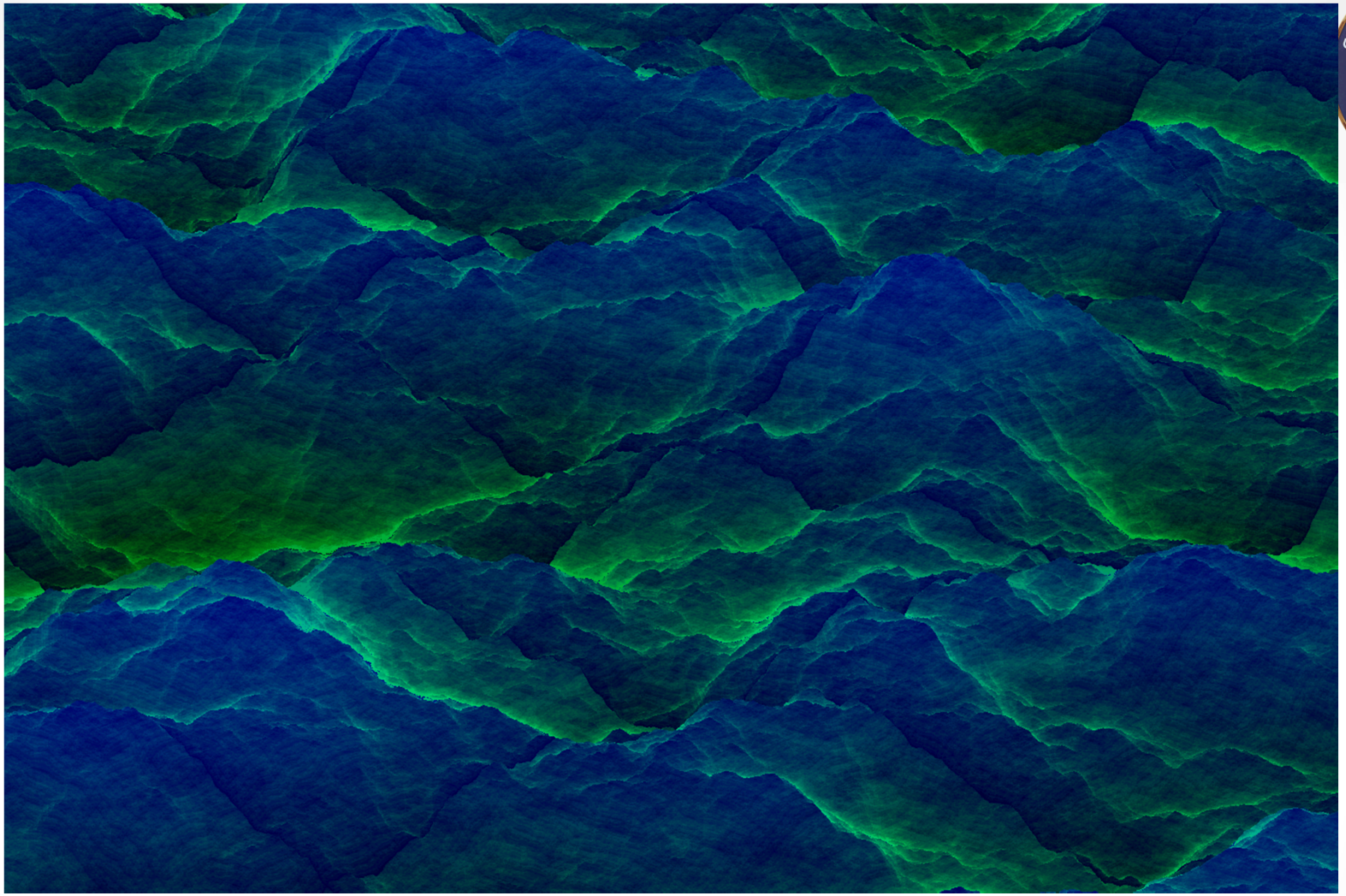


Bilinear Patch Intersection

- Intersect a ray $\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + t \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix}$ with a bilinear surface

  $z = A_{00} + A_{10}x + A_{01}y + A_{11}xy$
- $x, y \in [0,1]$
- Substitute ray equation into patch equation

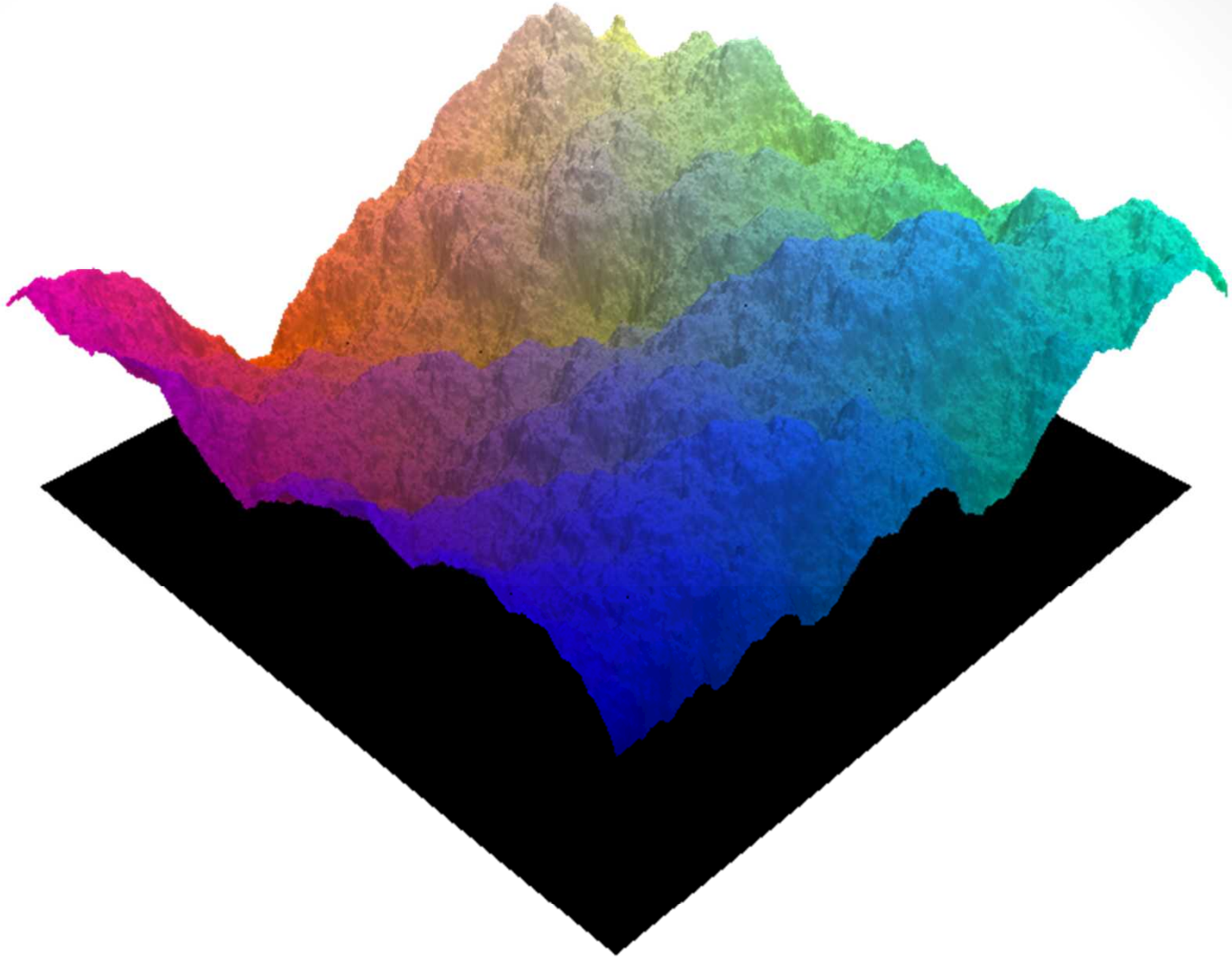
  $$Z_0 + td_z = A_{00} + A_{10}(X_0 + td_x) + A_{01}(Y_0 + td_y) + A_{11}(X_0 + td_x)(Y_0 + td_y)$$
- Solve for t $(A_{11}d_xd_y)t^2 + (A_{10}d_x + A_{01}d_y + A_{11}d_xy_0 + A_{11}d_yx_0 - d_z)t + (A_{00} - z_0) = 0$
  - Solve like sphere intersection
  - Make sure to stay inside of patch bounds
  - Ray can have 0, 1, or 2 intersection points



Results – Maximum Mipmap

Green channel used to indicate number of steps before surface intersection

Blue channel indicates height of terrain



Results – Bilinear Patch Intersection

Color represents intersection position

Shading based on patch normals

Questions?

References

- Schafhitzel, Tobias, Martin Falk, and Thomas Ertl. "Real-time rendering of planets with atmospheres." (2007).
- O'Neil, Sean. "Accurate atmospheric scattering." GPU Gems 2 (2005): 253-268.
- Dick, Christian, Jens Krüger, and Rüdiger Westermann. "GPU ray-casting for scalable terrain rendering." *Proceedings of EUROGRAPHICS*. Vol. 50. 2009.
- Tevs, Art, Ivo Ihrke, and Hans-Peter Seidel. "Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering." Proceedings of the 2008 symposium on Interactive 3D graphics and games. ACM, 2008.
- http://modis.gsfc.nasa.gov/ - MODIS Texture Data