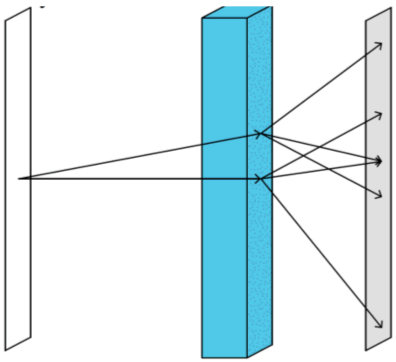
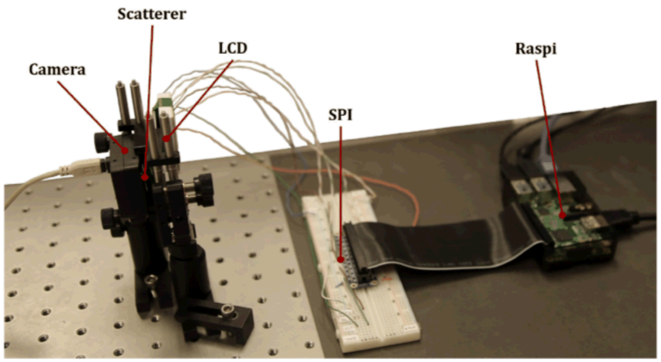
SAND2017-7270C





Lensless computational imaging using 3-D printed transparent elements

Charles F. LaCasse, Gabriel C. Birch, Amber L. Dagel, and Bryana L. Woo
Sandia National Laboratories

July 13 2017

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. SAND2011-0439P

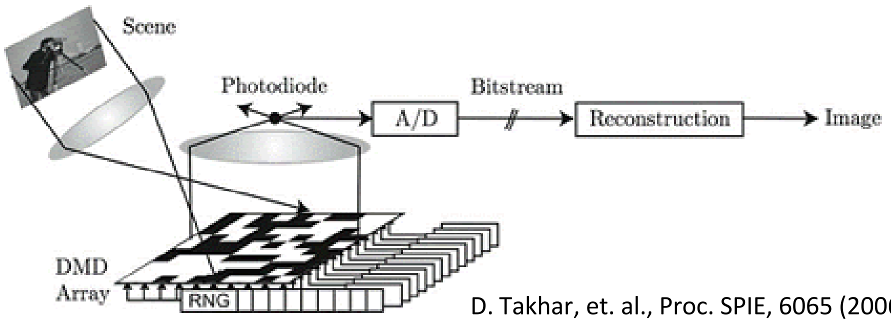
Introduction

1. Create 3-D printed scattering optical element
2. Calibration process to relate image space to object space
3. Simulate
4. Demonstrate prototype

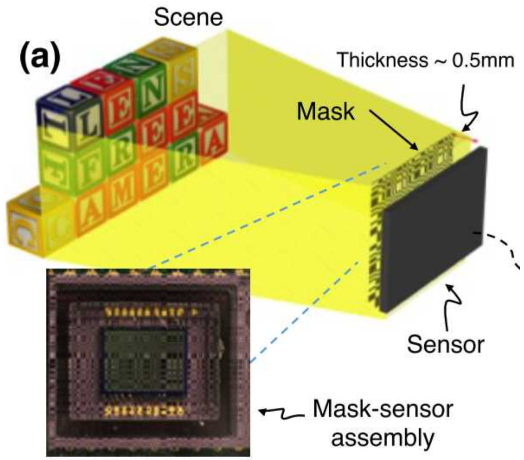
Background

- Computational Imagers
  - Single pixel camera
  - Random scattering compressive systems
  - Free form sensors
  - Foveating sensors
  - Coded aperture systems

R. Fergus, et. al., MIT CompSci & AI Lab (2006)



D. Takhar, et. al., Proc. SPIE, 6065 (2006)

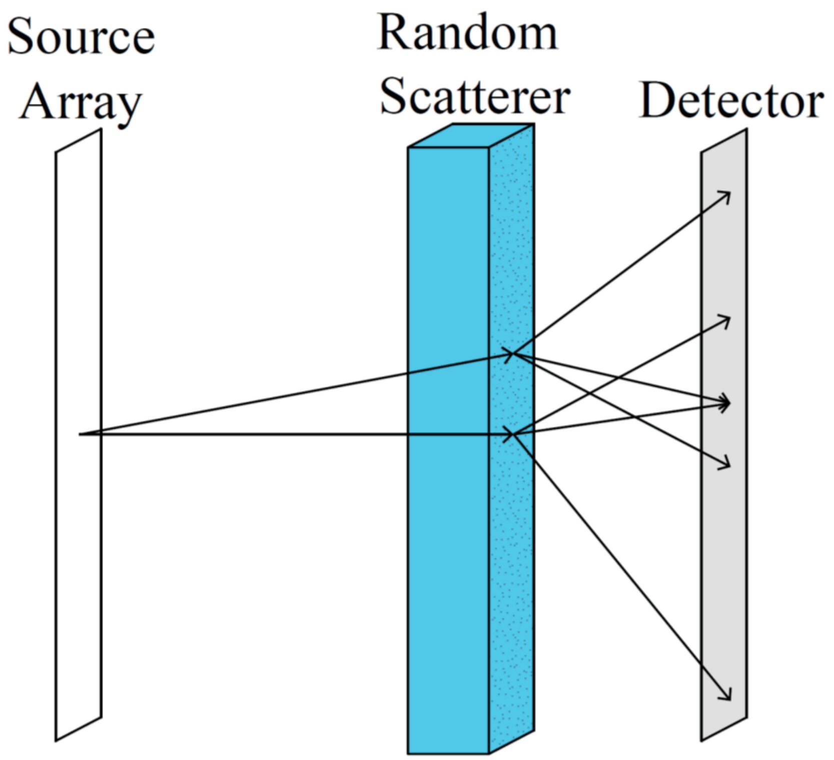


M.S. Asif, et. al., arXiv, 1-11 (2015)

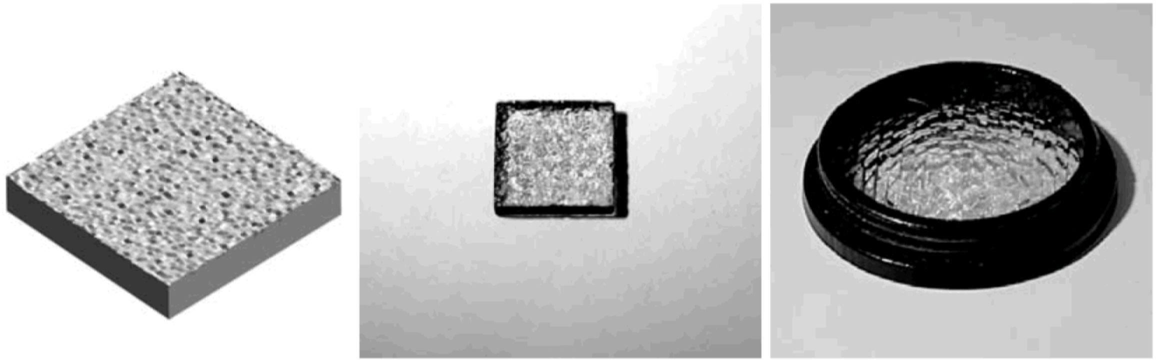
- Why do we care about computational imaging?
  - More variables to optimize, spread (or shift) complexity in a system
  - Potential to reduce size, weight, power consumption, and bandwidth

System Architecture

- Static lensless imaging system
- Three primary hardware components
  1. Source array
  2. Random scattering element
  3. Detector

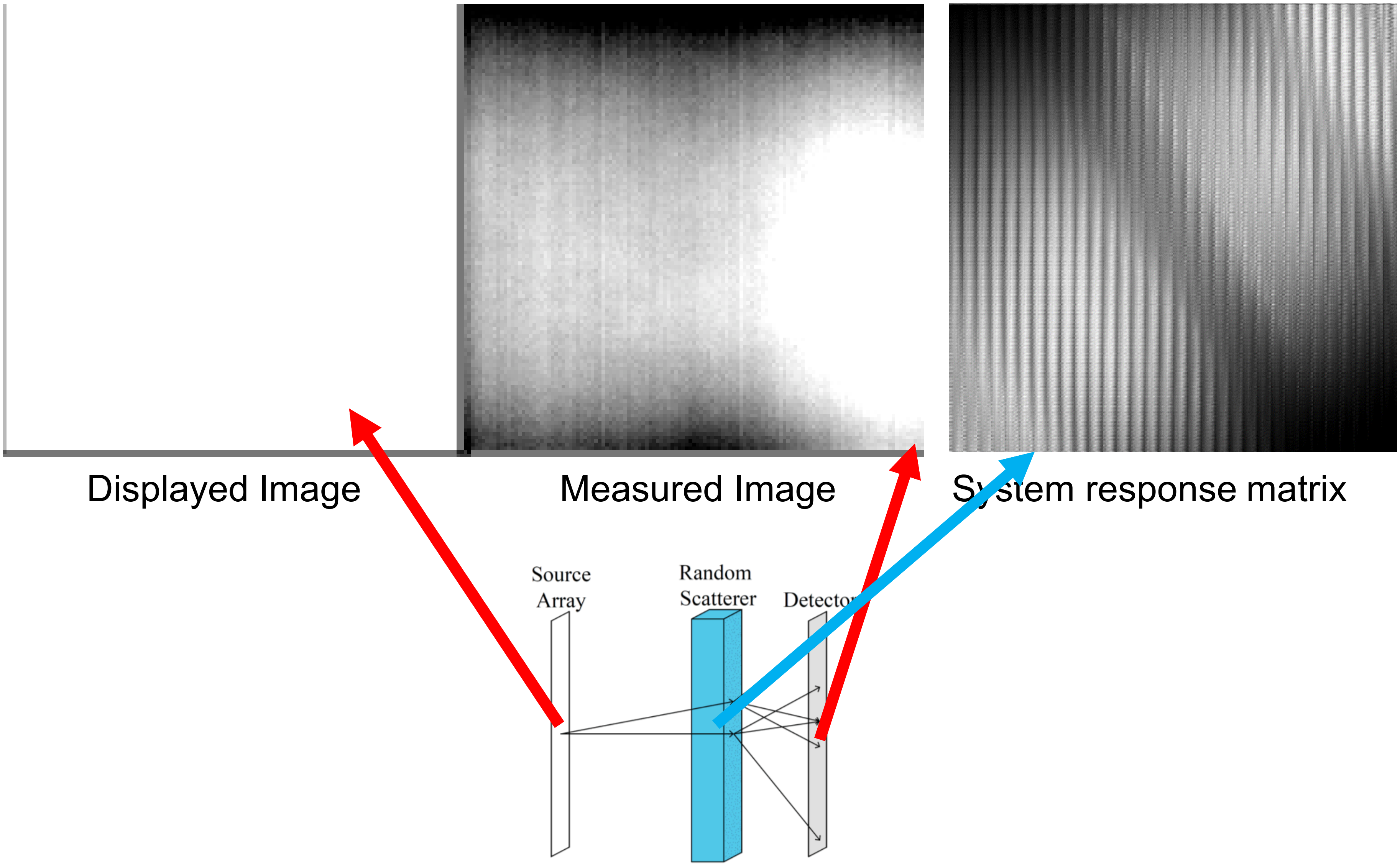


Random Scattering Element

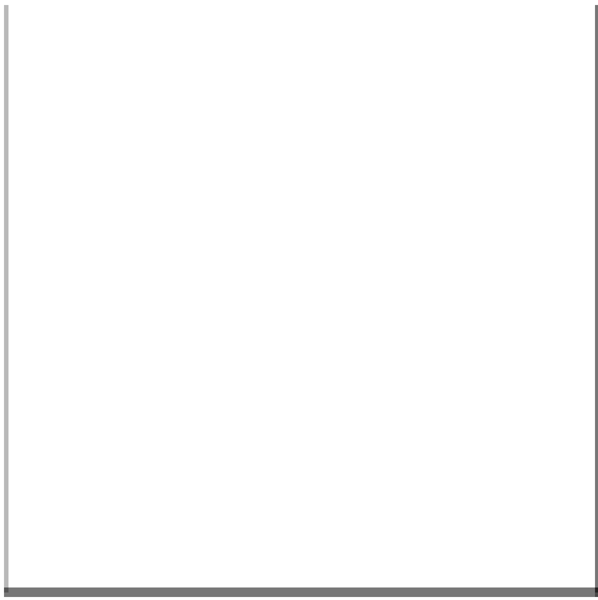


- Random scatterer can be created many ways
  - e.g., crystal growth, sand blasting, etc.
- CAD model of simple prism array
  - Facet angles from subdividing surface mesh of rectangular volume
  - Heights randomly varied
- Used an Object30 3-D printer and optically clear polymers to create both scatterer and opto-mechanics

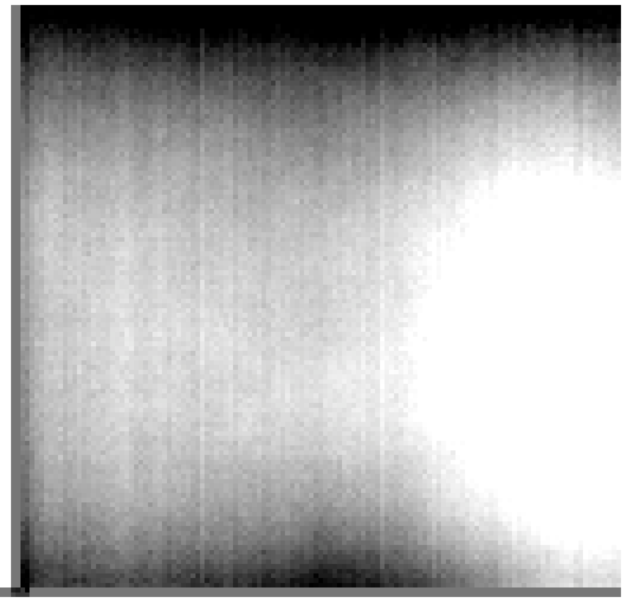
Calibration



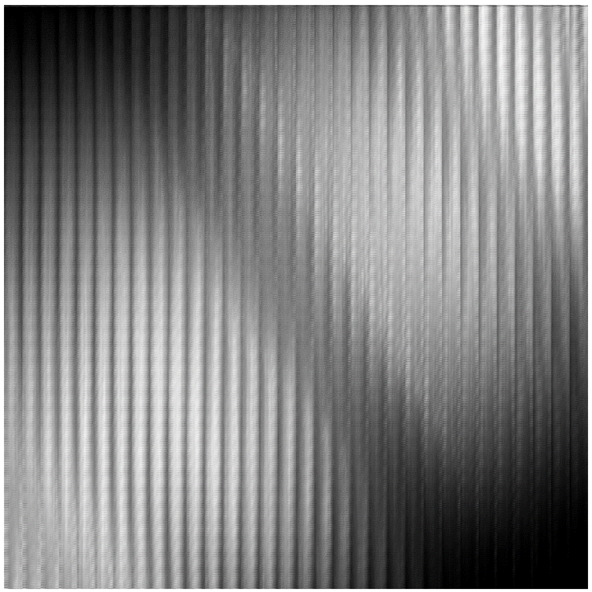
Calibration



Displayed Image



Measured Image



System response matrix

1. Build up transfer function definition by displaying series of images to system
2. Estimate system transfer function from measured data
3. Measure arbitrary scene by applying transfer function inverse to measured data to retrieve scene estimate

Calibration

- Solving basis pursuit denoising (BPDN) problem

$$\min \frac{1}{2} \left\| \mathbf{b} - \underline{\underline{\mathbf{X}}} \hat{\mathbf{T}}_{ij} \right\|_2^2 + \lambda \left\| \hat{\mathbf{T}}_{ij} \right\|_1$$

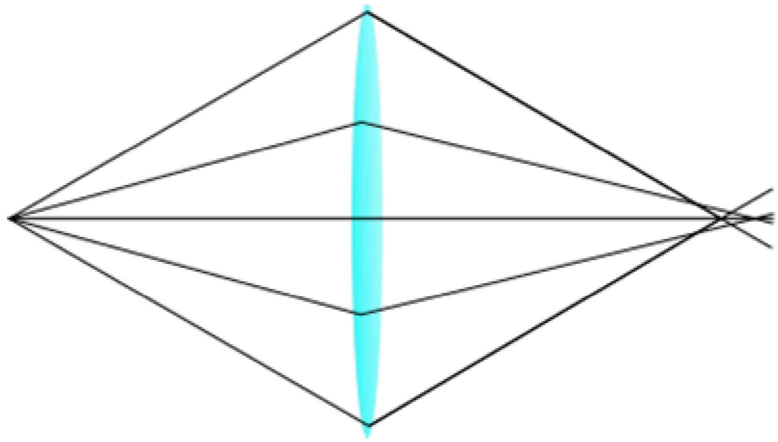
  - **b**: measured data,
  - **X**: calibration image
  - $\hat{\mathbf{T}}_{ij}^{-1}$: transfer function
  - λ is a weighting factor
- Estimate arbitrary scene using transfer function

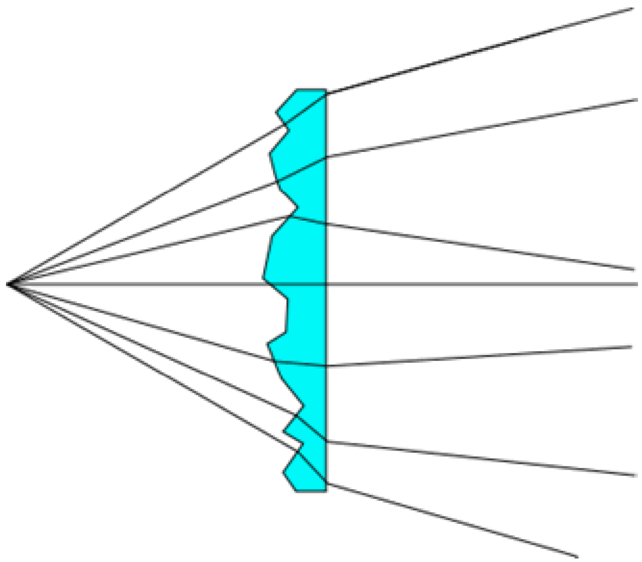
$$\hat{\mathbf{S}} = \hat{\mathbf{T}}_{ij}^{-1} \mathbf{b}$$

- Typically BPDN solves directly for reconstructed images
  - Here, solving for $\widehat{\boldsymbol{T}}$ and define **X** to be calibration array

Design Challenges

- Simulation is difficult
  - Need computational reconstruction to determine performance metric
  - Full simulation necessary to evaluate performance
  - Scattering surface needs to be adequately sampled for ray trace: much higher requirement than for smooth deterministic lens surfaces

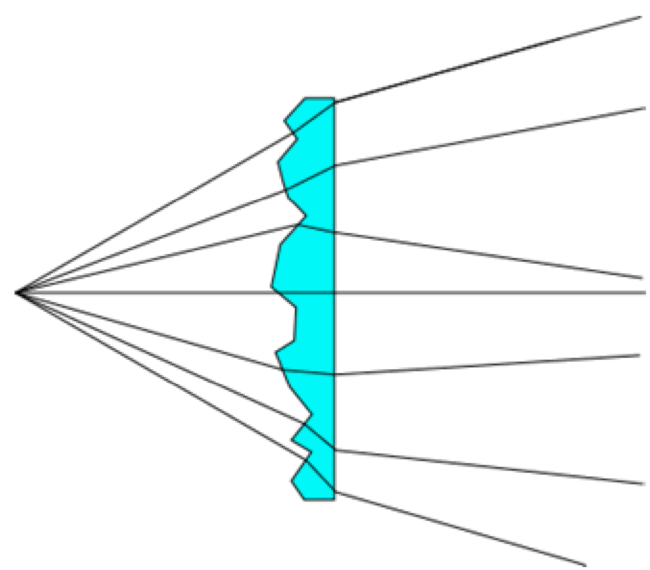




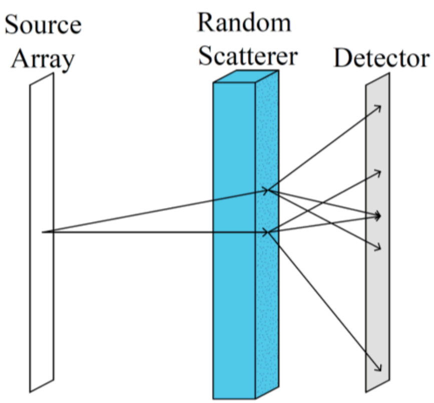
Raytrace Solution

- Traditional optical design tools are not well suited for this task
  - Example: 10^6 rays/image, 10^3 images/calibration for a single system
  - Very slow on CPU-based ray tracers
  - Moving to Monte Carlo simulations of multiple systems requires many CPUs or significant amount of time
- Solution: Nvidia OptiX ray trace API

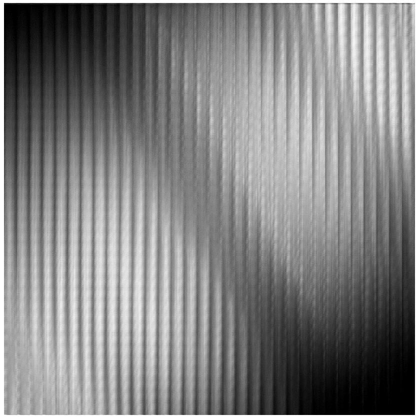
- Traditional optical design tools are not well suited for this task
  - Example: 10^6 rays/image, 10^3 images/calibration for a single system
  - Very slow on CPU-based ray tracers
  - Moving to Monte Carlo simulations of multiple systems requires many CPUs or significant amount of time
- Solution: Nvidia OptiX ray trace API

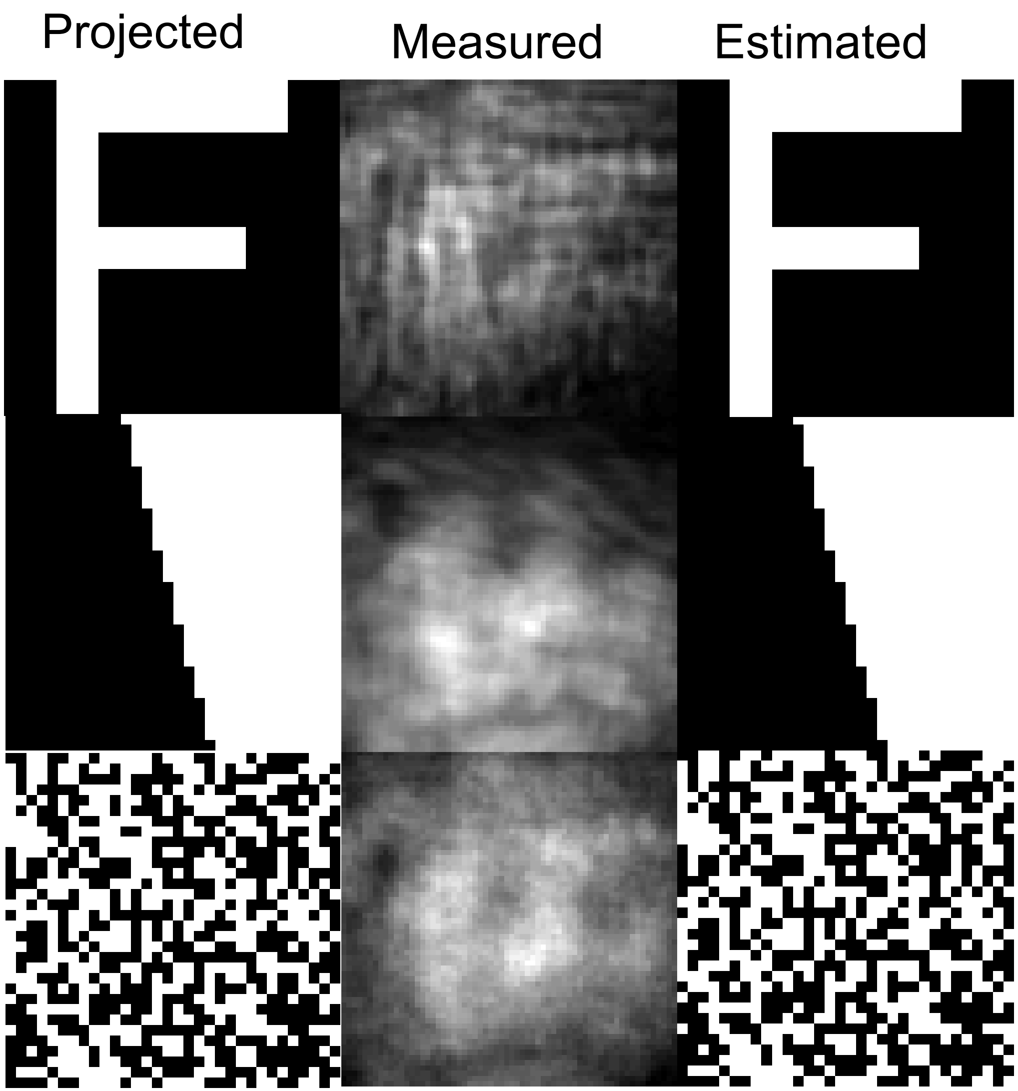


Results- Simulation



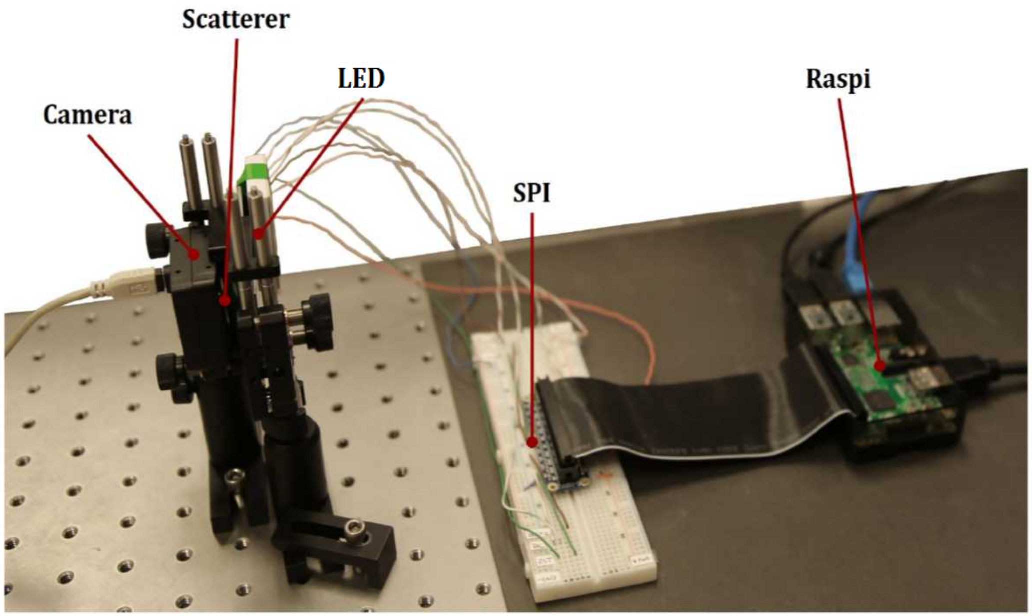
Calibration





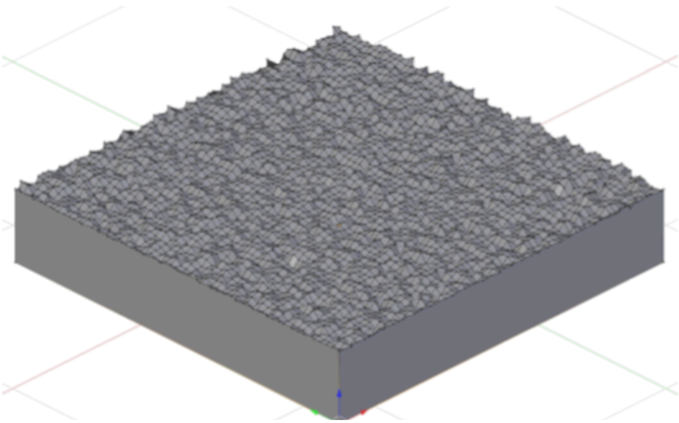
Prototype

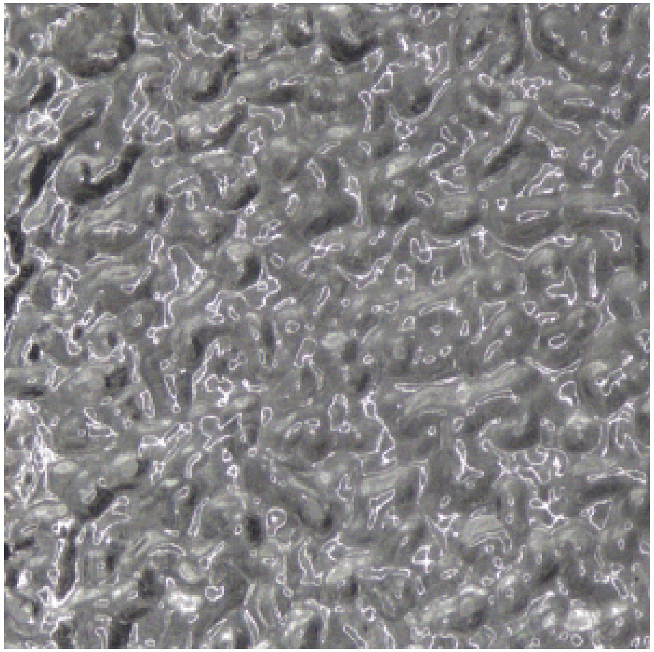
- Prototype:
  - Point Grey Chameleon 2 detector
  - 3-D printed scatterer
  - LED array

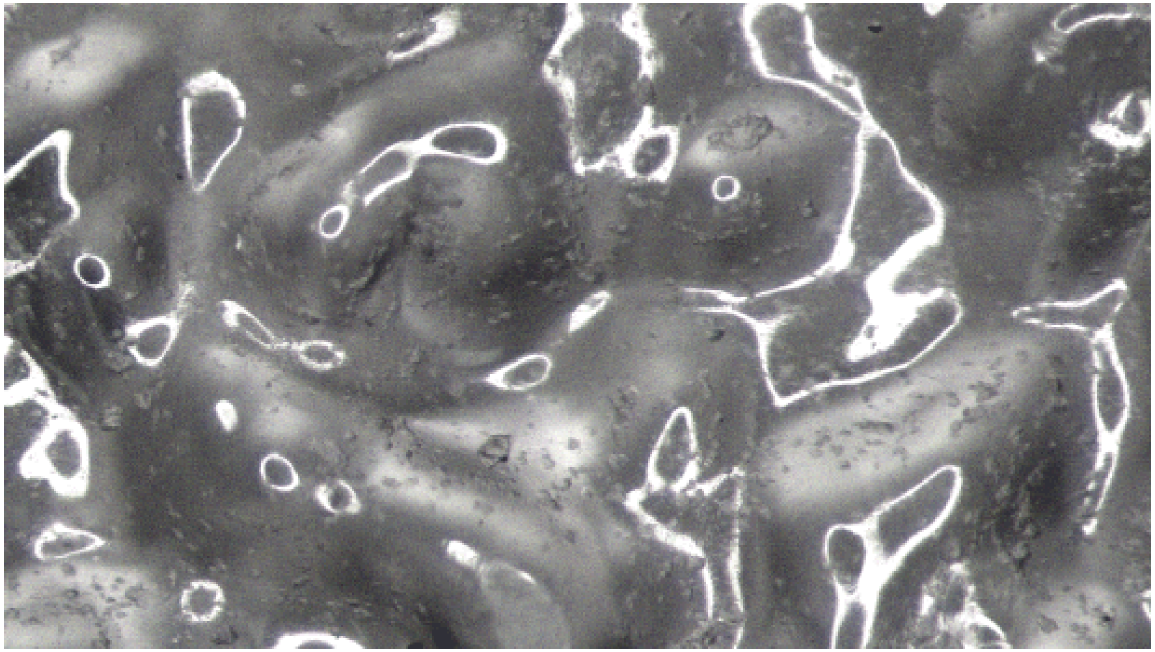


- Interface
  - LED controlled by Raspberry Pi 2
  - Camera controlled with MATLAB
  - Timing issues can be challenging
    - Master computer triggering the LED and Camera

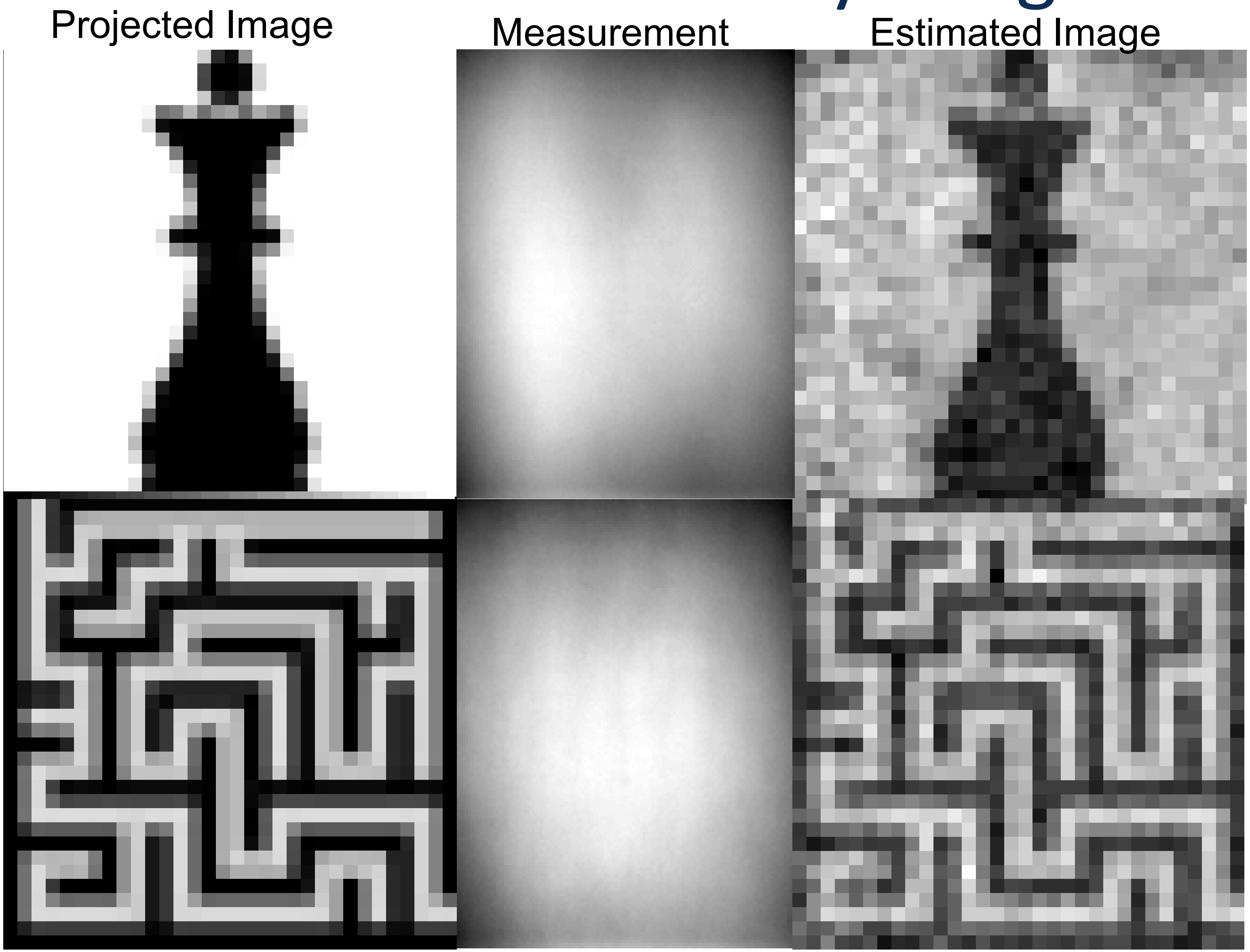
3-D printed element



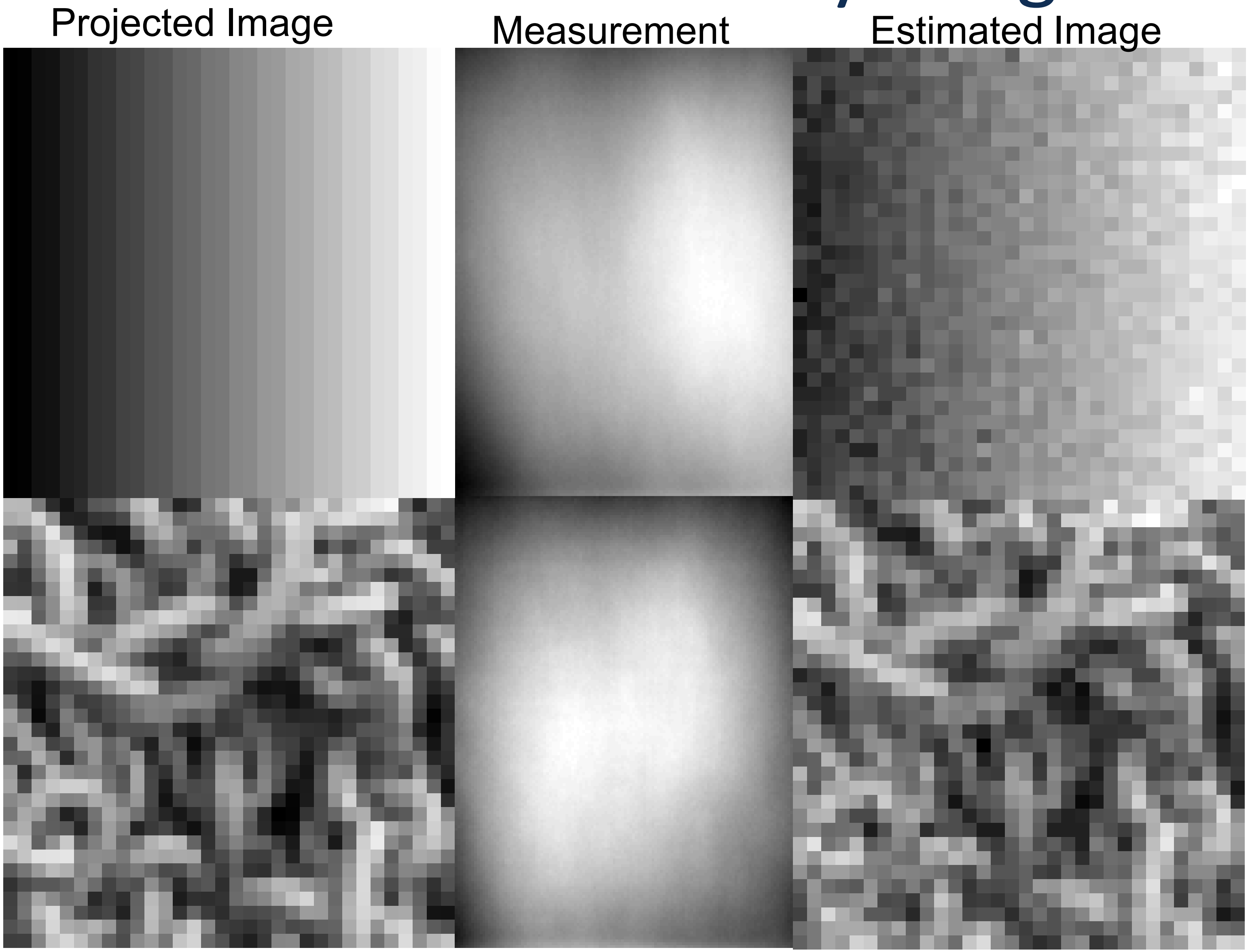




Results- Hardware Binary Images



Results- Hardware Binary Images



Conclusion

- Simulation, calibration, and demonstration of a lensless computational imaging system
- Reconstructions of computationally sensed raw data
- Inexpensive, off-the-shelf hardware
- Successfully reconstruct images from a system without a lens
  - 3-D printed optical element: random, low optical quality material, uncontrolled surface quality, inexpensive

Questions?

Back up slides

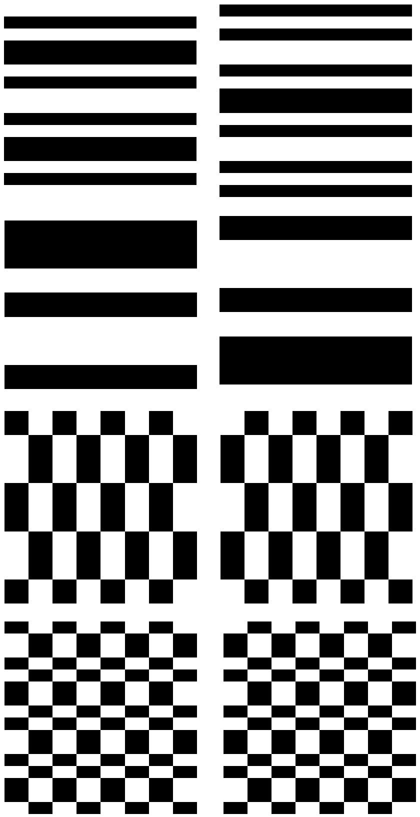
Hadamard Realization-

- Hadamard matrix calibration pattern chosen to assume orthonormality on the measurement matrix
  - Hadamard matrix ranges [-1, 1]
  - Cannot display negative intensity in physical hardware
- Split Hadamard matrix into positive and negative mappings

$$\underline{\underline{\mathbf{H}}}^{+} = \begin{cases} 1 & \underline{\underline{\mathbf{H}}} > 0 \\ 0 & \underline{\underline{\mathbf{H}}} < 0 \end{cases} \qquad \underline{\underline{\mathbf{H}}}^{-} = \begin{cases} 1 & \underline{\underline{\mathbf{H}}} < 0 \\ 0 & \underline{\underline{\mathbf{H}}} > 0 \end{cases}$$

$$\underline{\underline{\mathbf{X}}} = \underline{\underline{\mathbf{H}}}^{-} - \underline{\underline{\mathbf{H}}}^{+}$$

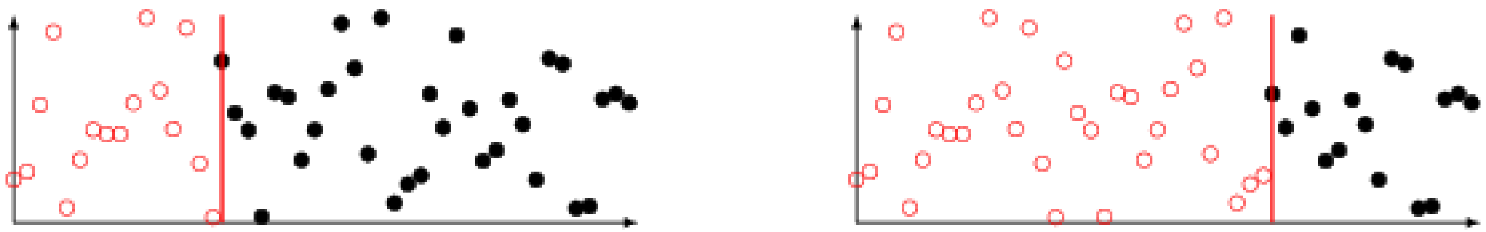
$$\mathbf{b} = \mathbf{b}^{+} - \mathbf{b}^{-}$$



Quantitative system quality metrics

- Quantitative metric needed to evaluate performance of reconstructed images
- Binary test targets can be analyzed with Bit Error metric

$$\text{Bit Error} = \sum_{n,m=0}^{N,M} \frac{\text{Test Image}(n,m) \oplus (\text{Reconstructed Test Image}(n,m) > T)}{(N \times M)}$$



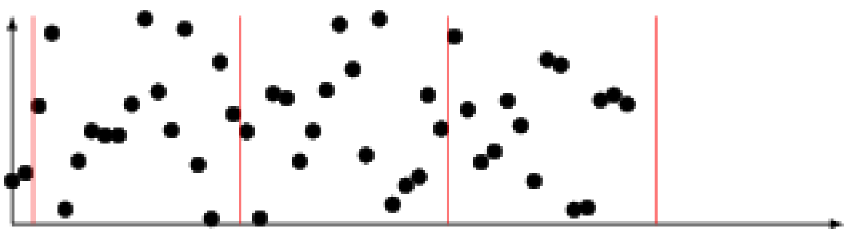
Scanning threshold

- Grayscale images require a more complex metric

$$\text{Scaled offset image} = \begin{cases} 0 & \alpha \times \text{Reconstructed Test Image} + x_o \leq 1 \\ i & i \leq \alpha \times \text{Reconstructed Test Image} + x_o < i + 1 \\ I & I \leq \alpha \times \text{Reconstructed Test Image} + x_o \end{cases}$$

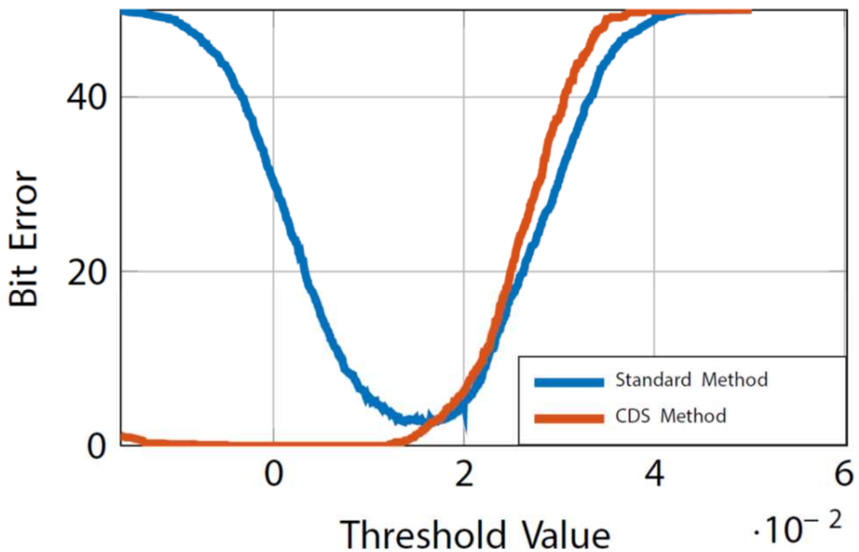
$$\text{Percent Error} = \text{mean}\left(\frac{\text{Test Image}(n,m) - \text{Scaled offset image}(n,m)}{I}\right)$$

- Maximum pixel value I used as normalization factor
- Grayscale images scan and shift digitization bins

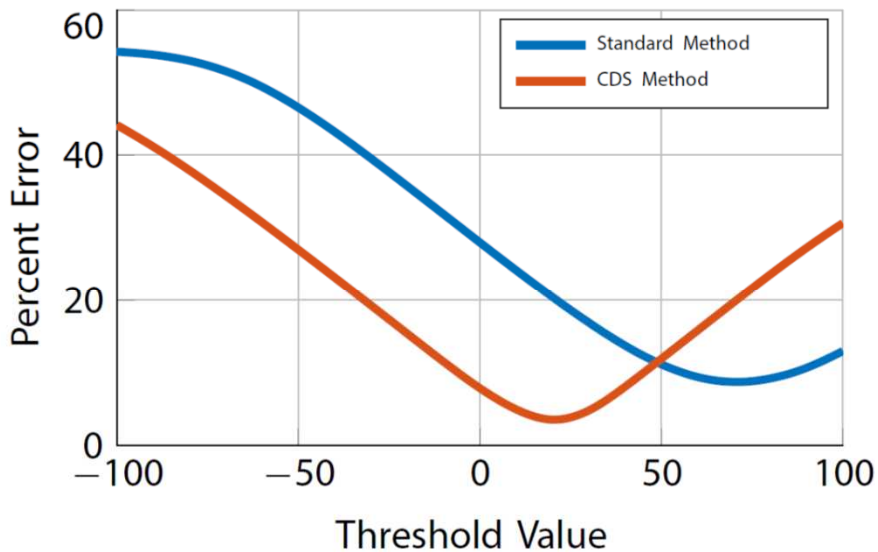


Results- Bit Error and Percent Error

Binary images:
Scan threshold, calculate Bit Error



Greyscale images:
Scan bin width & offset, find Percent Error



- Standard reconstruction method shows clear threshold range for minimum bit error and percent error
- Correlated double sampling (CDS) reduces error significantly in both cases