Remaining course logistics

- Final project reports are due on Tuesday 12/14 at 23:59 ET.
  - Gradescope submission site available.

- Keep an eye out for a Piazza announcement on returning borrowed equipment.

Class evaluation*s* – please take them!

- CMU's Faculty Course Evaluations (FCE): https://cmu.smartevals.com/
- TA evaluation: https://www.ugrad.cs.cmu.edu/ta/F21/feedback/
- 15-463/663/862 end-of-semester survey: https://docs.google.com/forms/d/e/1FAIpQLSc6eXw_tbcxIF7-fm882V_7g80Q-l_34oObplTMCqOUgMcEkw/viewform
- Please take all three of them, super helpful for developing future offerings of the class.
- Thanks in advance!

Today’s judges



Matthew O’Toole



Jun-Yan Zhu

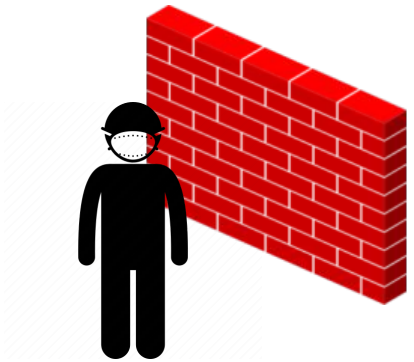
Computational Periscopy with an Ordinary Camera

BY: JOSH ABRAMS

Imagine...

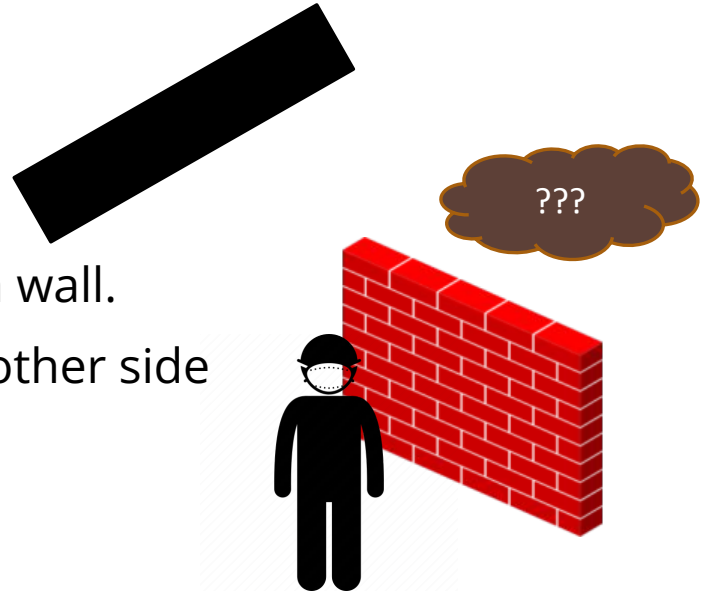


1. You are standing near a corner with a wall.



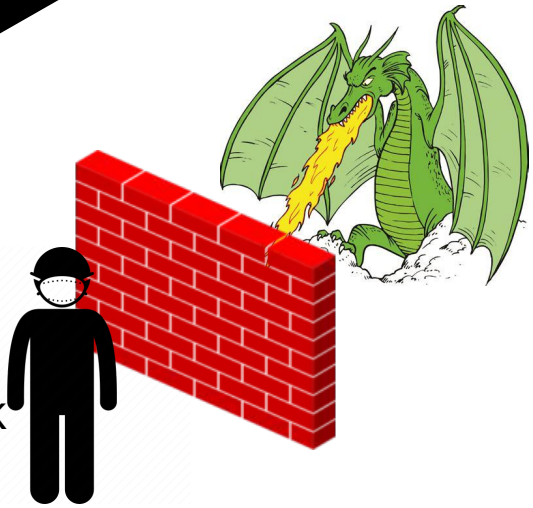
Imagine…

1. You are standing near a corner with a wall.
2. There’s something really cool on the other side that you want to take a picture of.

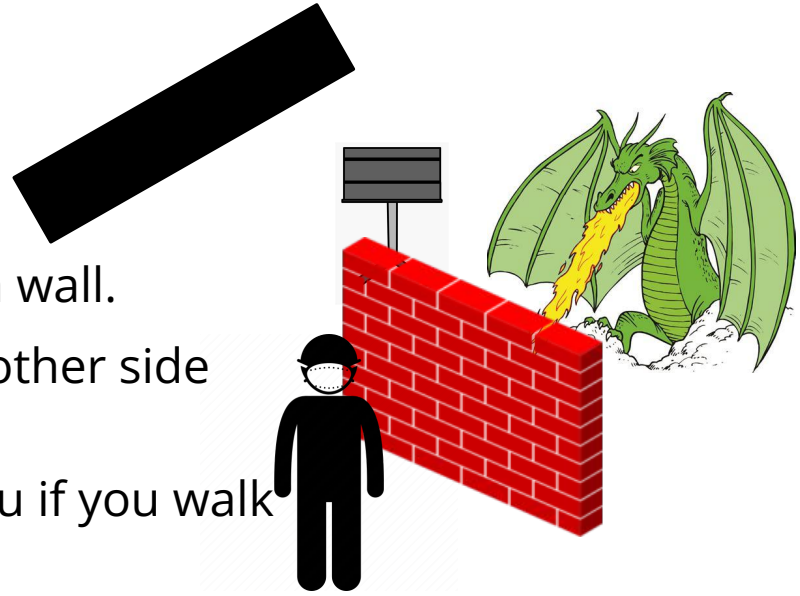


Imagine...

1. You are standing near a corner with a wall.
2. There's something really cool on the other side that you want to take a picture of.
3. The thing on the other side will kill you if you walk around the corner.

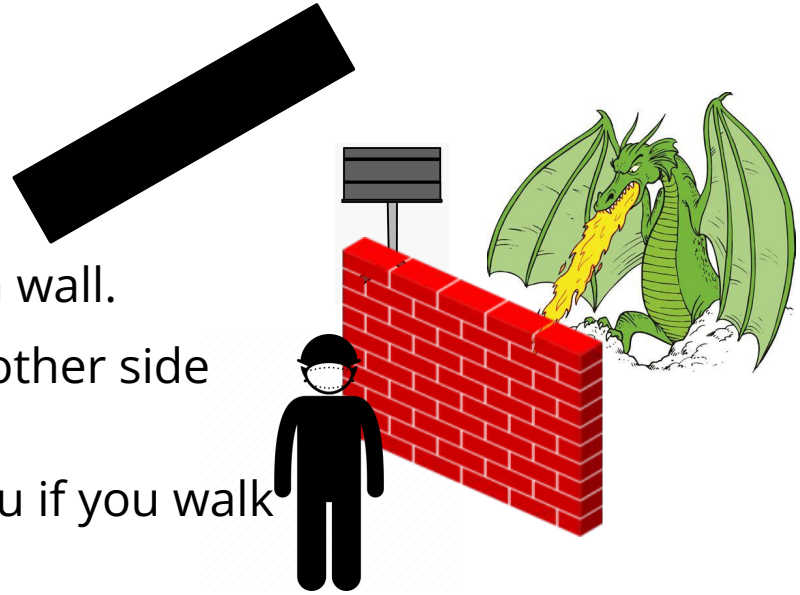


Imagine…



1. You are standing near a corner with a wall.
2. There’s something really cool on the other side that you want to take a picture of.
3. The thing on the other side will kill you if you walk around the corner.
4. There’s something on the other side with known shape (but potentially unknown position).

Imagine…



1. You are standing near a corner with a wall.
2. There's something really cool on the other side that you want to take a picture of.
3. The thing on the other side will kill you if you walk around the corner.
4. There's something on the other side with known shape (but potentially unknown position)

LETTER

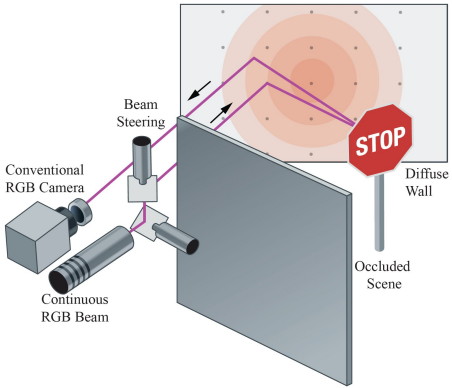
https://doi.org/10.1038/s41586-018-0868-6

Computational periscopy with an ordinary digital camera

Charles Saunders[1,2], John Murray-Bruce[1,2] & Vivek K Goyal[1]*

Sounds awfully contrived don’t’cha think?

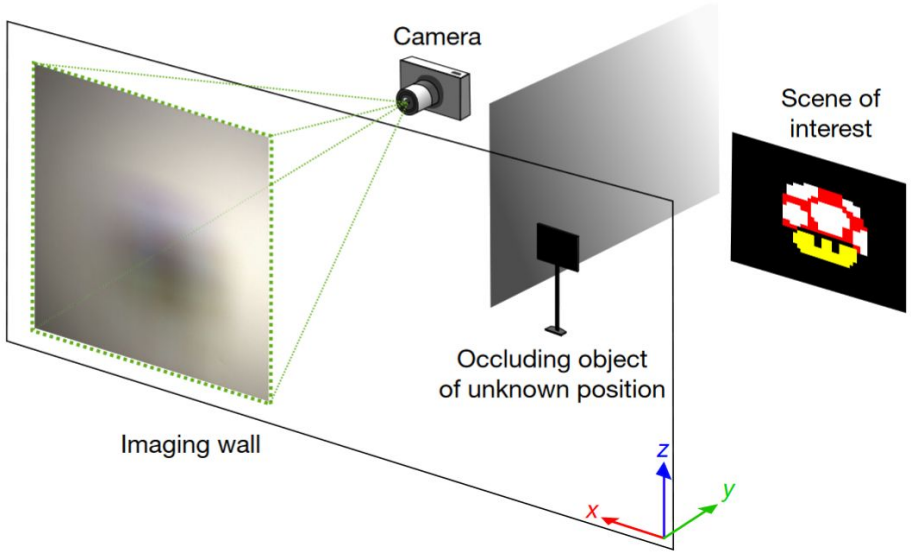
1. Well… yes
2. But potentially less contrived and a step forward in some ways
3. Maybe there’s something you could throw around the corner





Alright, so how does this work?

Alright, so how does this work?



[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

Alright, so how does this work?

1. Make assumptions to simplify the modified rendering equation

$$I(\boldsymbol{p}_{\mathrm{w}}) = \int_{\boldsymbol{x}\in S} \left\{ \frac{\cos[\measuredangle(\boldsymbol{p}_{\mathrm{w}} - \boldsymbol{x}, \boldsymbol{n}_x)]\cos[\measuredangle(\boldsymbol{x} - \boldsymbol{p}_{\mathrm{w}}, \boldsymbol{n}_{\mathrm{w}})]}{\left|\left|\boldsymbol{p}_{\mathrm{w}} - \boldsymbol{x}\right|\right|_2^2} \times V(\boldsymbol{x}; \boldsymbol{p}_{\mathrm{w}}; \boldsymbol{p}_{\mathrm{o}})\mu(\boldsymbol{x}, \boldsymbol{p}_{\mathrm{w}})f(\boldsymbol{x}) \right\} \mathrm{d}\boldsymbol{x} + b(\boldsymbol{p}_{\mathrm{w}})$$

$$(1) \longrightarrow \boldsymbol{y} = A(\boldsymbol{p}_{\mathrm{o}})\boldsymbol{f} + \boldsymbol{b}$$

[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

Alright, so how does this work?

2. Estimate the occluder position and use this to estimate the lightfield matrix

$$\hat{\boldsymbol{p}}_{\mathrm{o}} = \underset{\boldsymbol{p}_{\mathrm{o}}}{\operatorname{argmax}} \left|\left| A(\boldsymbol{p}_{\mathrm{o}})[A(\boldsymbol{p}_{\mathrm{o}})^{\mathrm{T}} A(\boldsymbol{p}_{\mathrm{o}})]^{-1} A(\boldsymbol{p}_{\mathrm{o}})^{\mathrm{T}} \boldsymbol{y} \right|\right|_2^2 \qquad (2)$$

[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

Alright, so how does this work?

3. Solve least squares problem with total variance regularization

$$\hat{\boldsymbol{f}} = \underset{\boldsymbol{f}}{\operatorname{argmin}} \left\| \hat{A}\boldsymbol{f} - \boldsymbol{y} \right\|_2^2 + \lambda \left\| \boldsymbol{f} \right\|_{\mathrm{TV}} \tag{3}$$

$$TV(f) = \int_{\Omega} |\nabla f|$$

[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

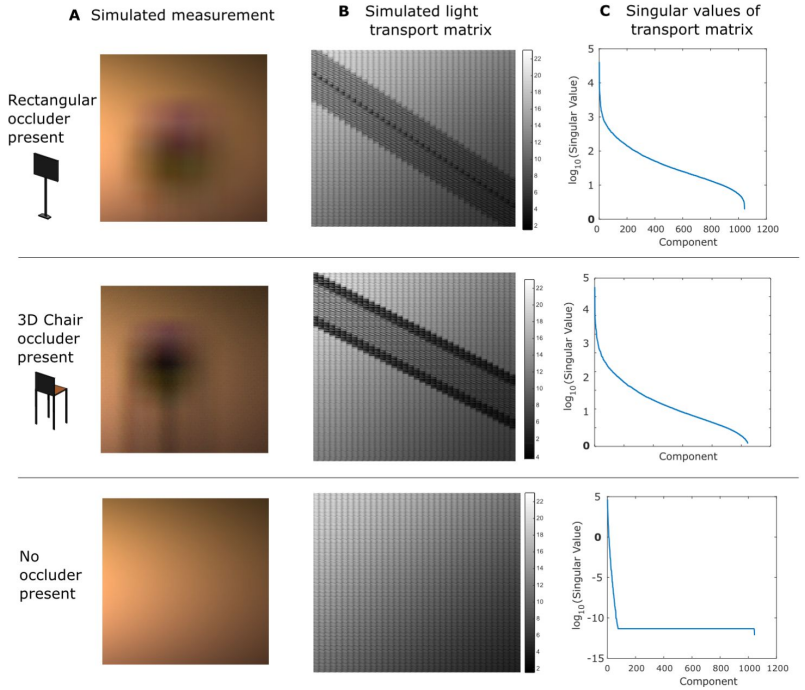


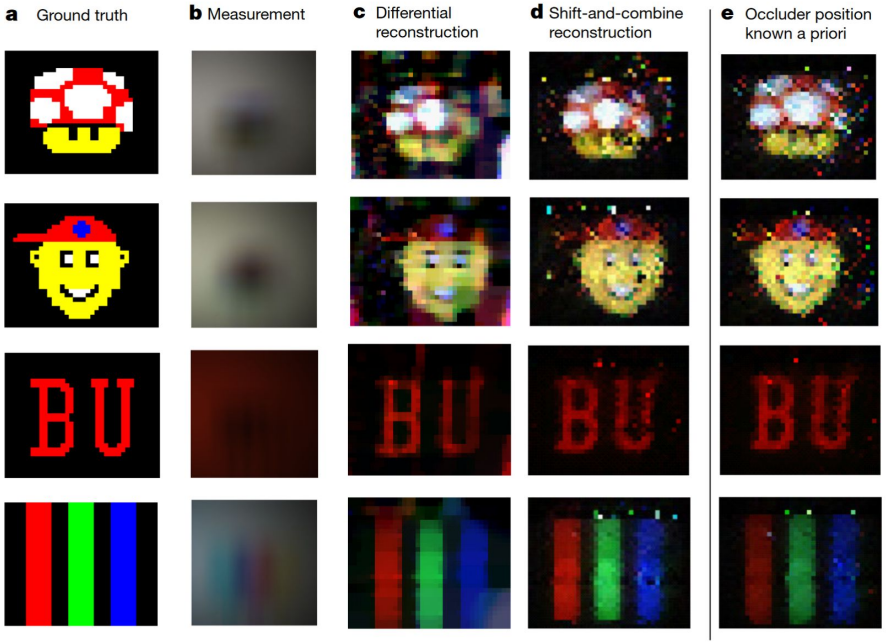
Figure S4: **The effect of the occluder on the conditioning of the transport matrix.** A simulated camera measurement using the Mushroom ground truth (**A**), the simulated transport matrix $\mathbf{A}(\boldsymbol{p}_o)$ (**B**), and the singular values of this transport matrix (**C**).

Why the occluder?

- Improves conditioning for inversion of the light transport matrix. Without it, the columns are rather smooth and similar since every point on the wall receives light from every point in the image. This results in a not-very-well-defined inverse, as shown at left.

[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

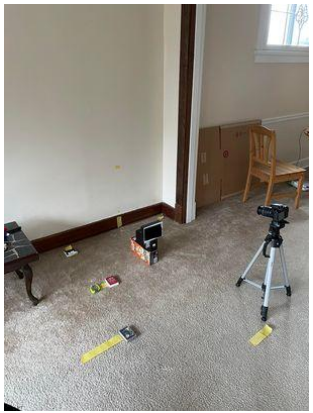
Did it work for them?



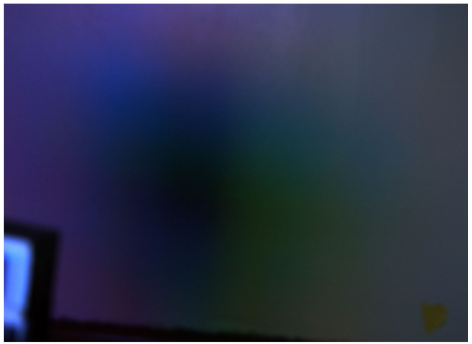
[1] Charles Saunders, John Murray-Bruce, and Vivek K Goyal. Computational periscopy with an ordinary digital camera. *Nature*, 565(7740):472–475, 2019.

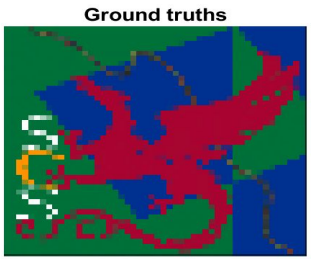
Did it work for you?

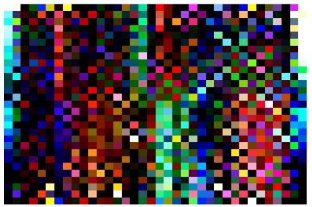
No :(— Will rant if there's time













Thanks! Any Questions?

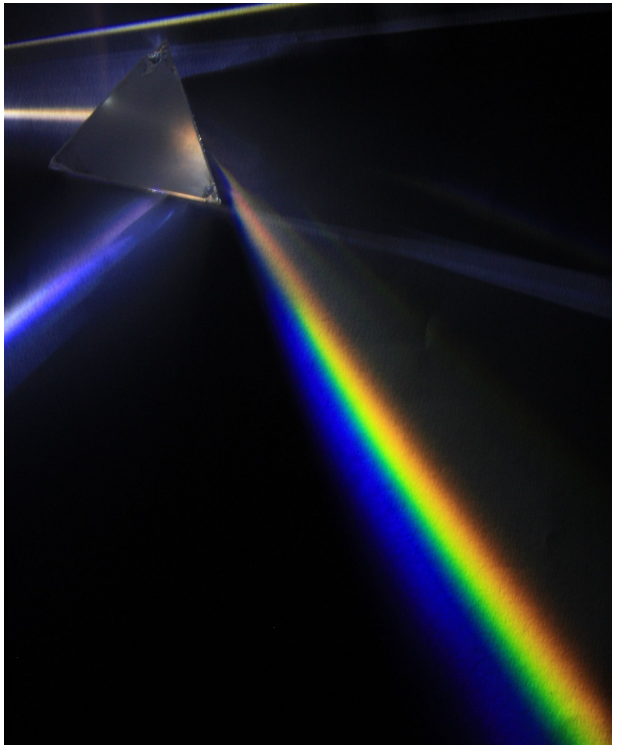
Citations

- All professional-looking equations and figures come from:
- [1] Saunders, C., Murray-Bruce, J., & Goyal, V. K. (2019). Computational periscopy with an ordinary digital camera. *Nature*, *565*(7740), 472–475. https://doi.org/10.1038/s41586-018-0868-6

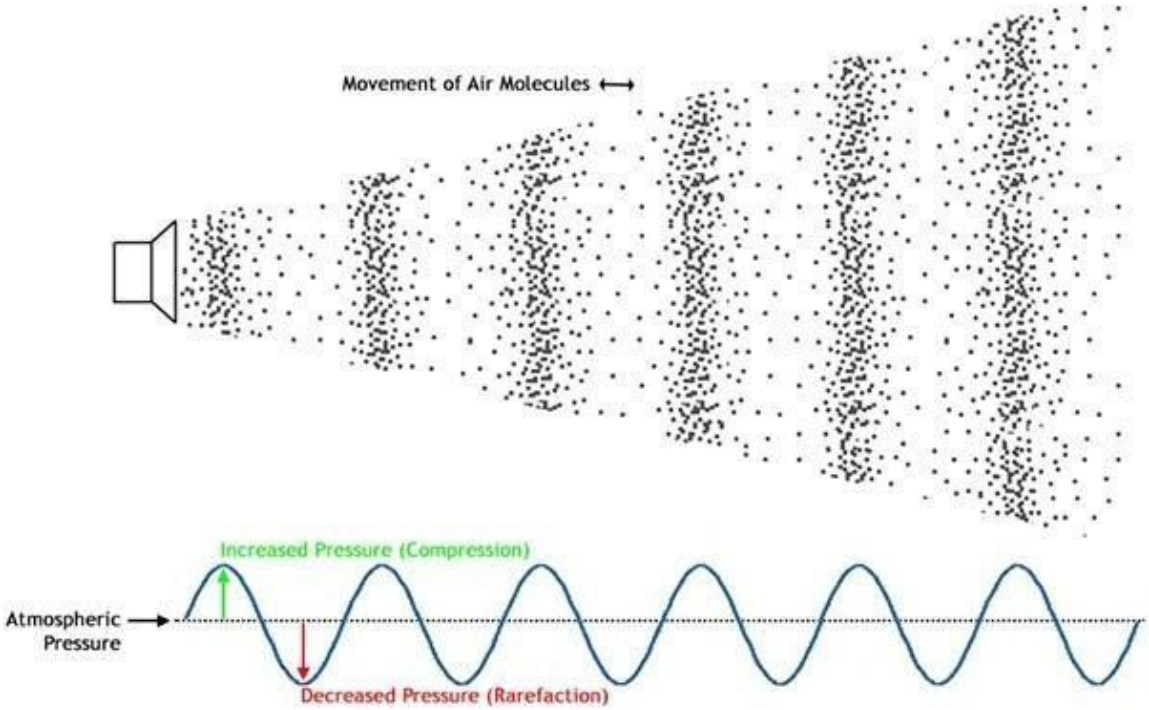
Acoustic lenses
(Deblurring weird lenses)

Hossein Baktash

Optics (EM waves)



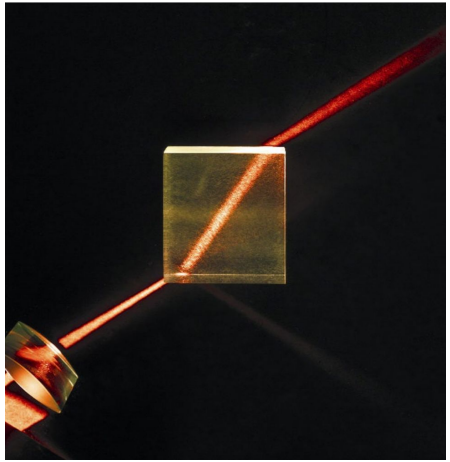
Acoustics (mechanical waves)

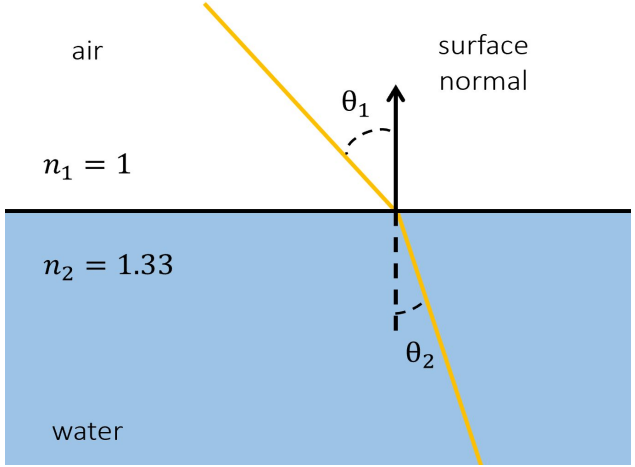


Acoustics + Optics ??



Refraction



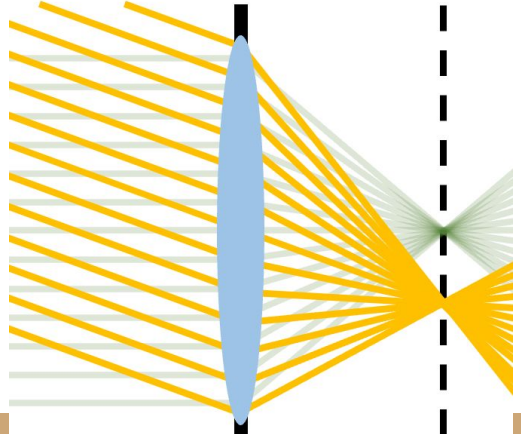


Snell's law

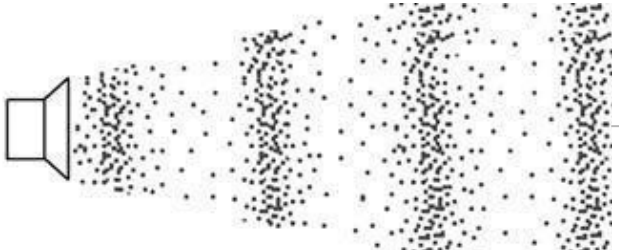
$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

Depends on the material

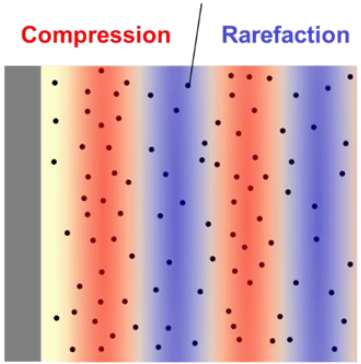
- And that's how lenses work



What do acoustic waves have to do with this?



Pressure in the material changes (spatially and temporally)



This changes refractive index at every point

Too hard!

$$\frac{n^2-1}{n^2+2} = \frac{4\pi\rho_c N_A}{3M}\alpha$$

Let's use this one **just for water**

$$n = 1.333 + cp$$

Where c is a constant and p is pressure

Change in refraction

- How the change happens: $n = 1.333 + cp$
- c has been empirically found for water and it is very small!

$$c = 1.402 \times 10^{-5}$$

- e.g. to make a change of 0.1 in refractive index of water, we need

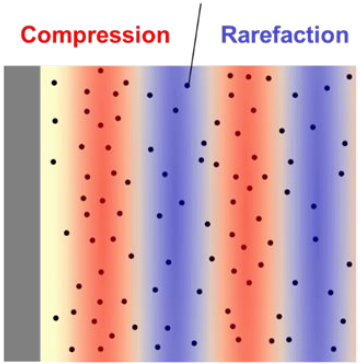
$$p \sim 2000\,MPa$$

- Remember glass (~1.5) vs air (~1.0)

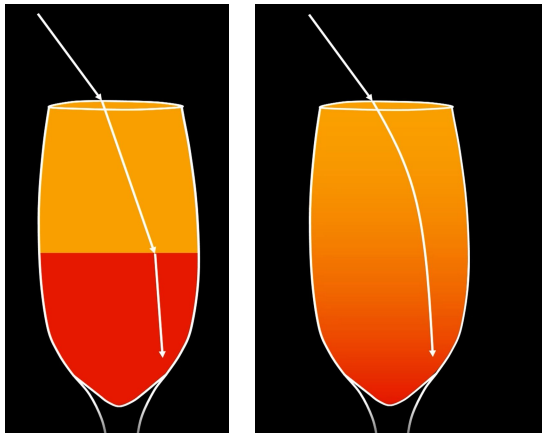
So, it's hard to affect light with mechanical waves



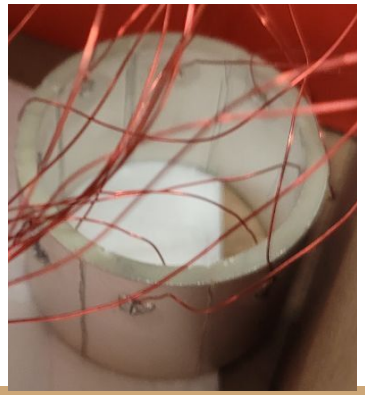
But it's possible!



Continuous refraction:

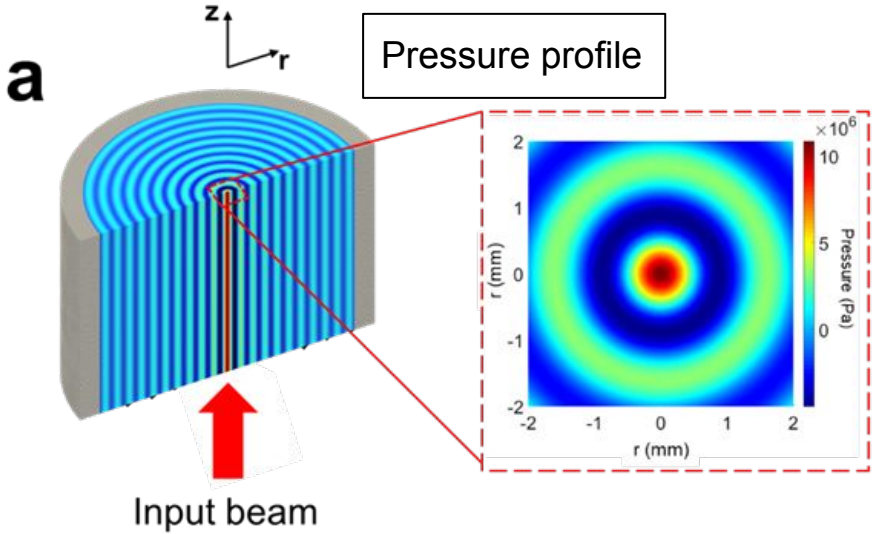


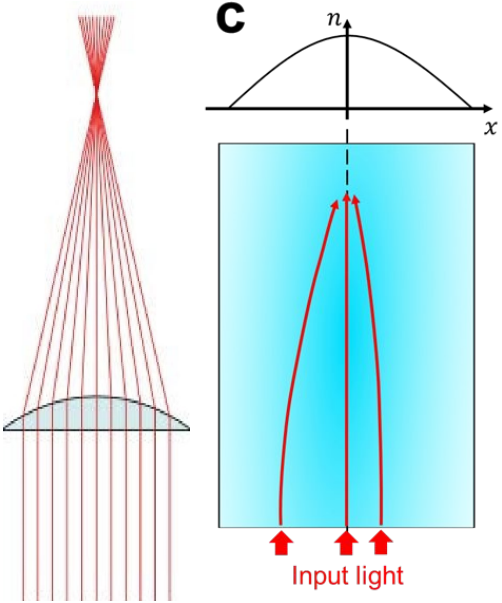
Ignore the wires!



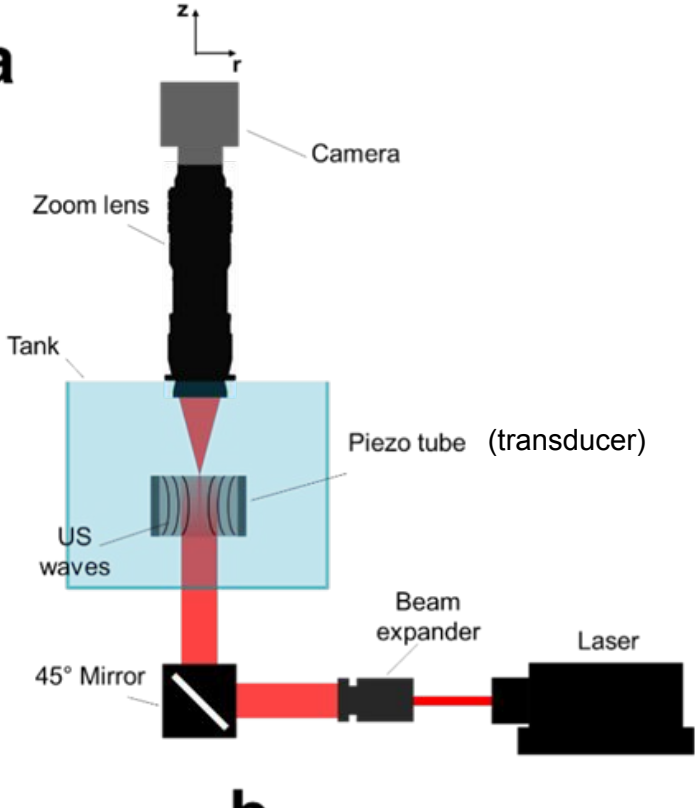
- Cylindrical transducer
- Just apply voltage to the surface

Pressure profile

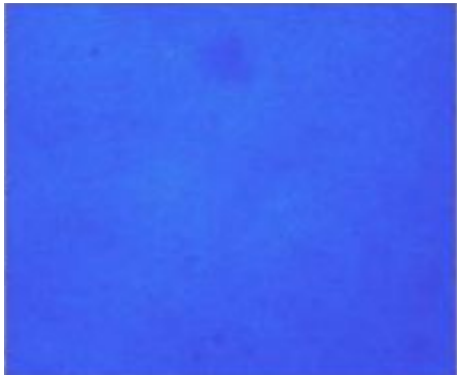




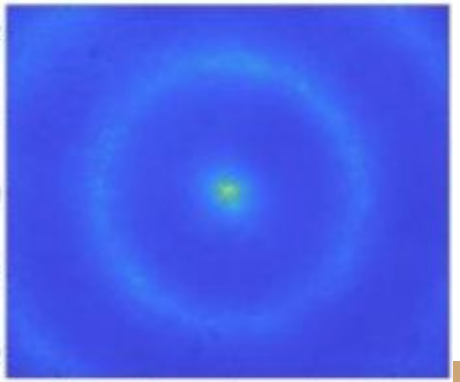
c
n
x
Input light


a
z
r
Camera
Zoom lens
Tank
Piezo tube
(transducer)
US
waves
Beam
expander
Laser
45° Mirror

US OFF

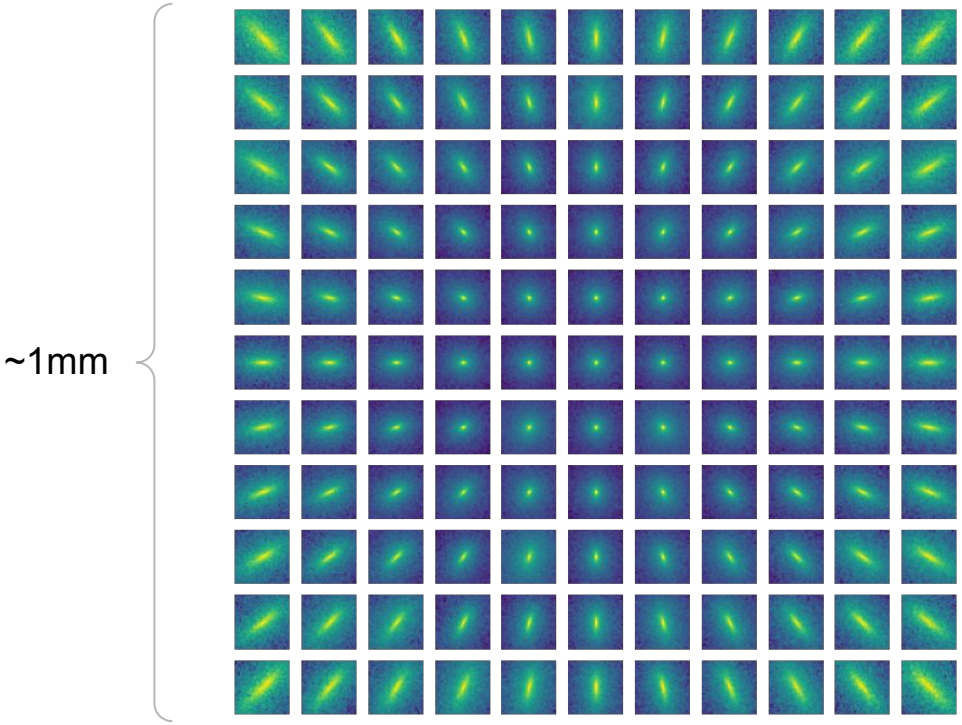


US ON

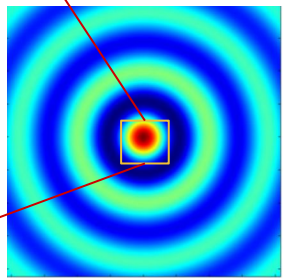


Can we do imaging with this?

- Yes. Here are the PSFs of our lens



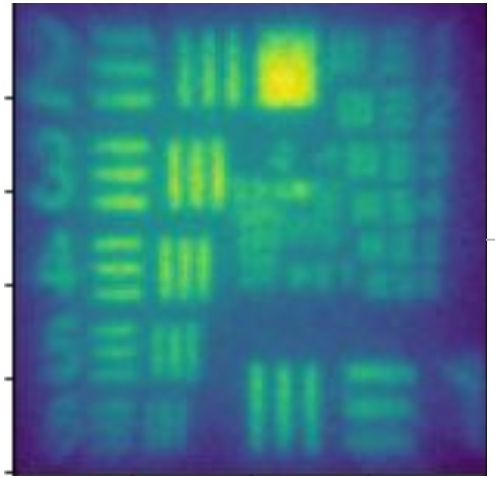
- Spatially varying PSF
- Almost Gaussian
- Deconvolution is very doable
- Can do sparse sampling of PSFs and treat them as locally invariant PSFs

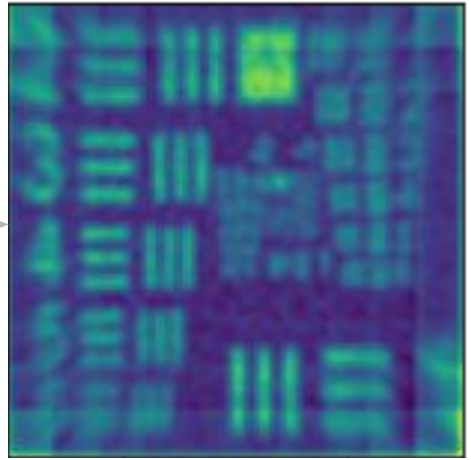


Pressure pattern

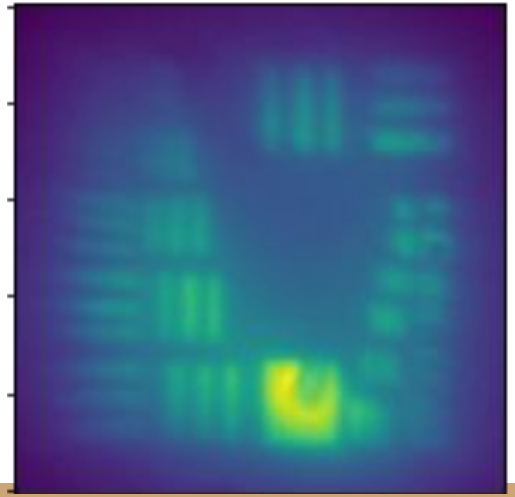
Results

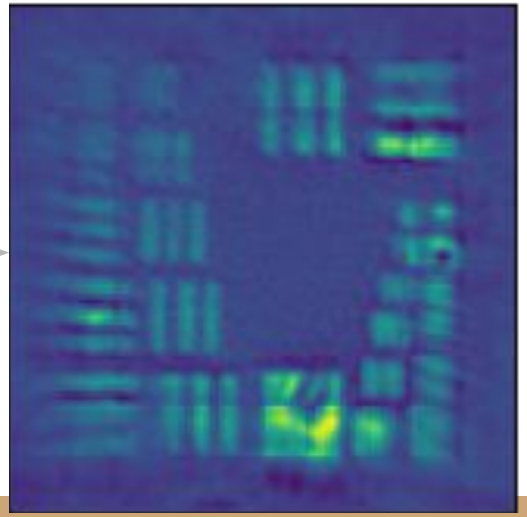
Simulated data



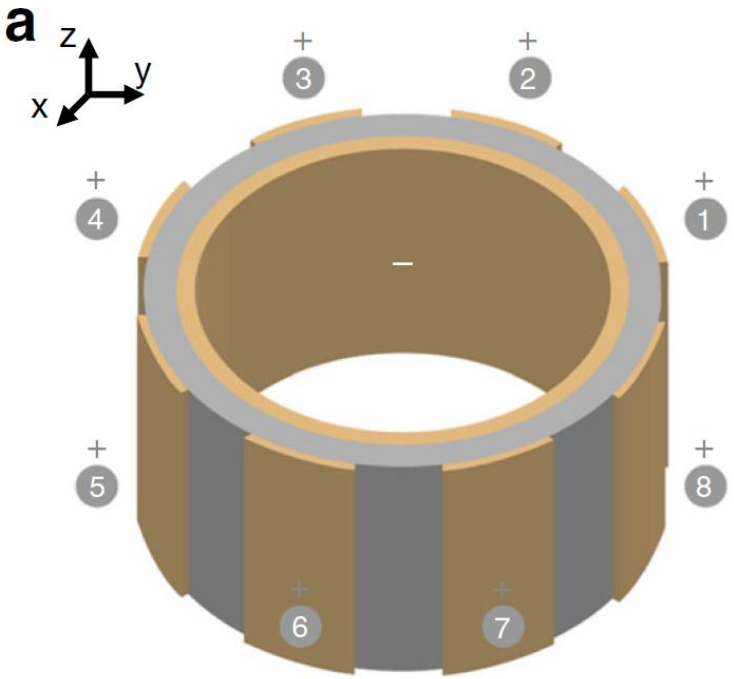


Experimental data

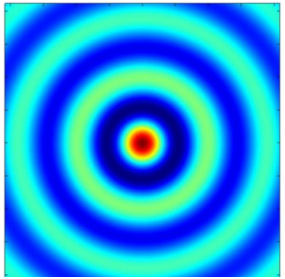




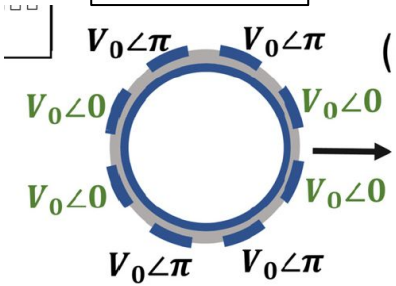
More interesting patterns

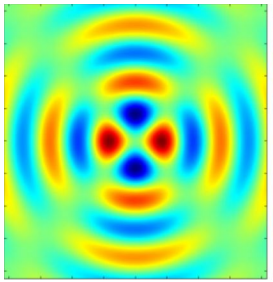


Mode 0 pattern:



Mode 2:

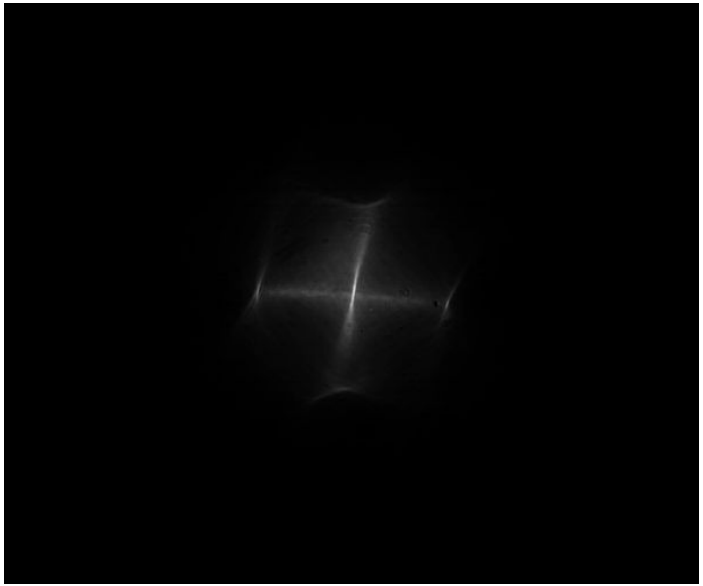




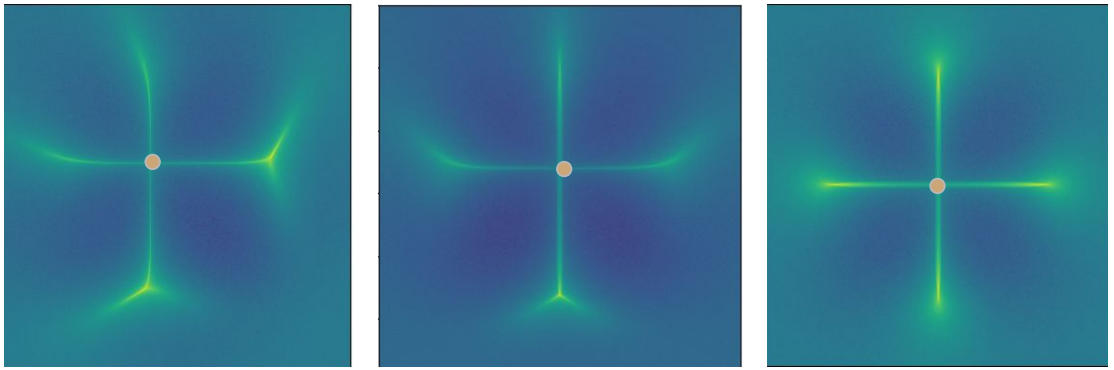
Mode 2

- Why? Because it can be extended to travelling wave (not a closed cylinder)

Beam pattern in Mode 2



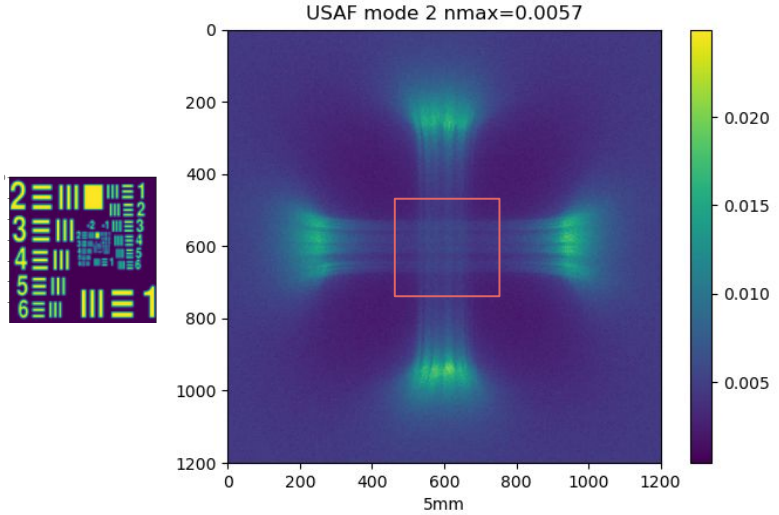
3 PSFs



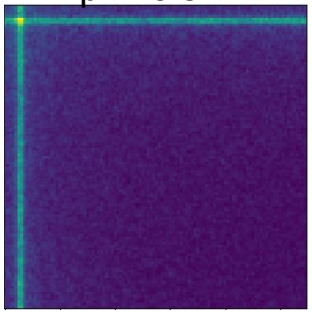
PSFs are to large to deconvolve!

Ignore the outer parts

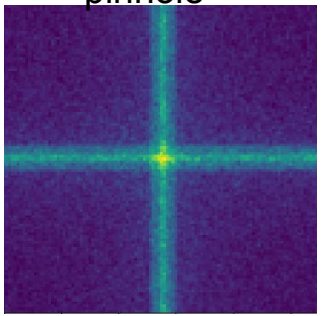
An image under mode 2



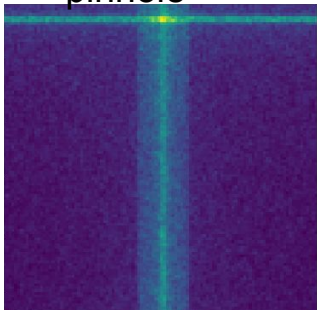
Top left pinhole

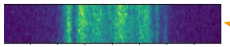


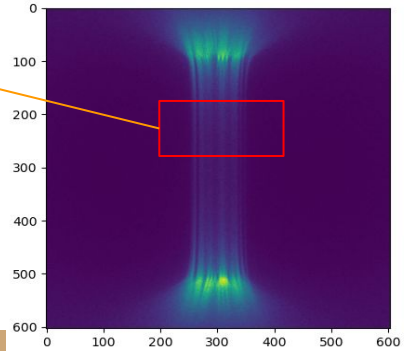
Center pinhole



Top center pinhole



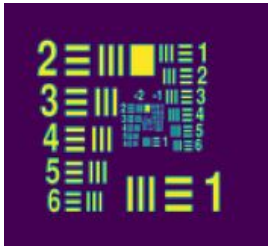


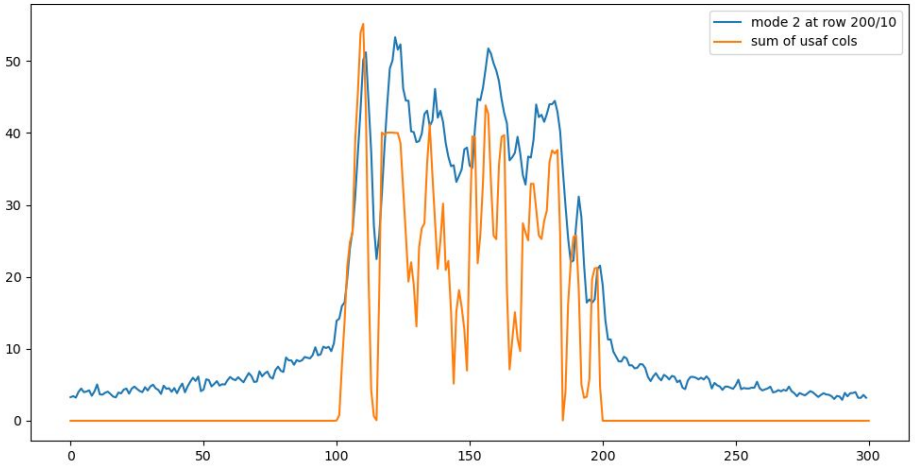


Pulsing light in sync with ultrasound gives u this:

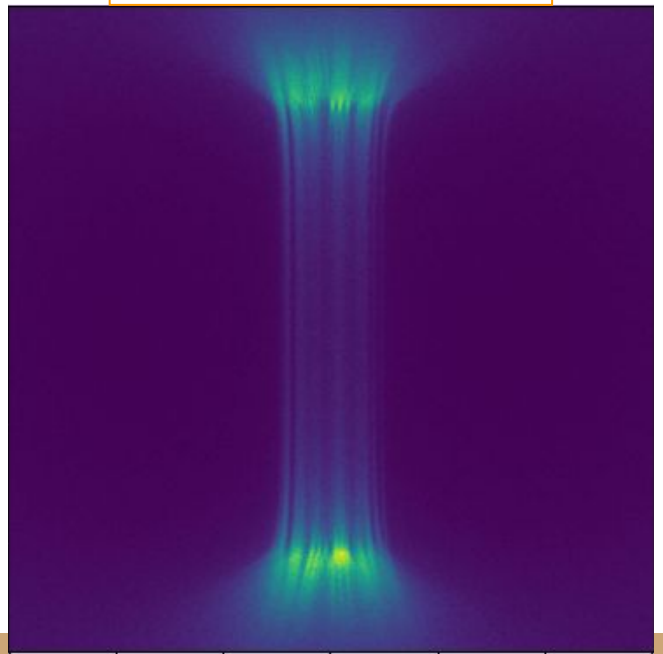
The inner part takes line integrals only

Project all pixels on this line



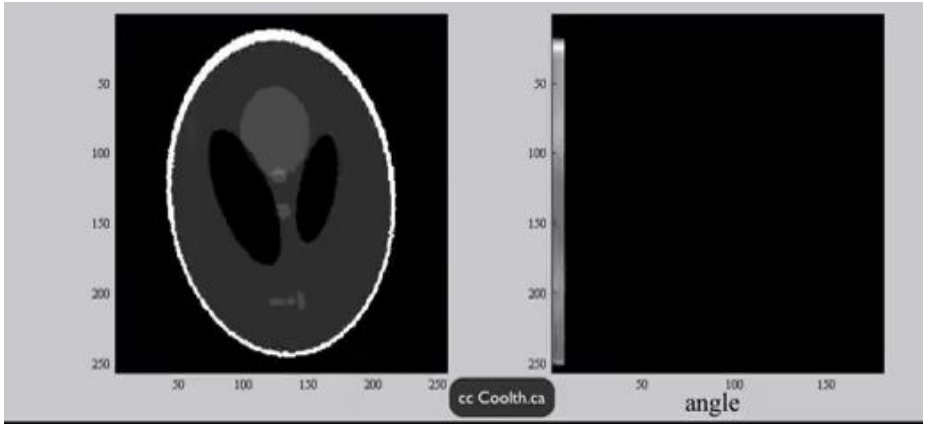


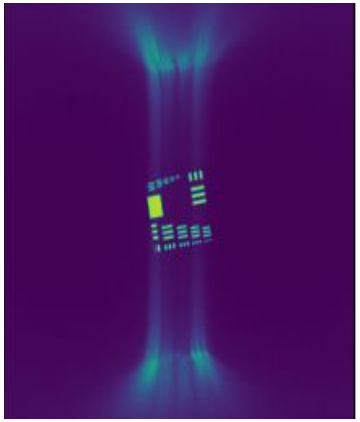
- Orange plot is kinda being copied in every row
- Take an average over rows
- This also helps with noise

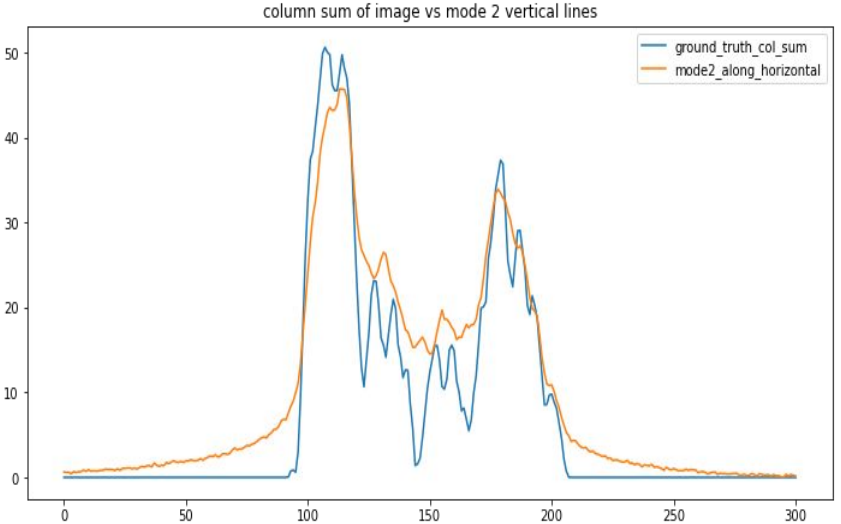


Radon transform

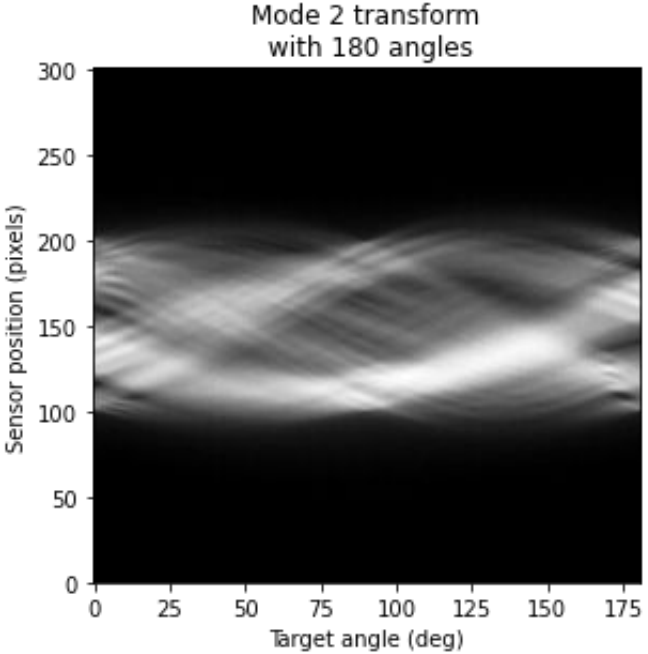
- Project to 1D
- Rotate the image a little
- Project to 1D
- Rotate
- Project
- ...

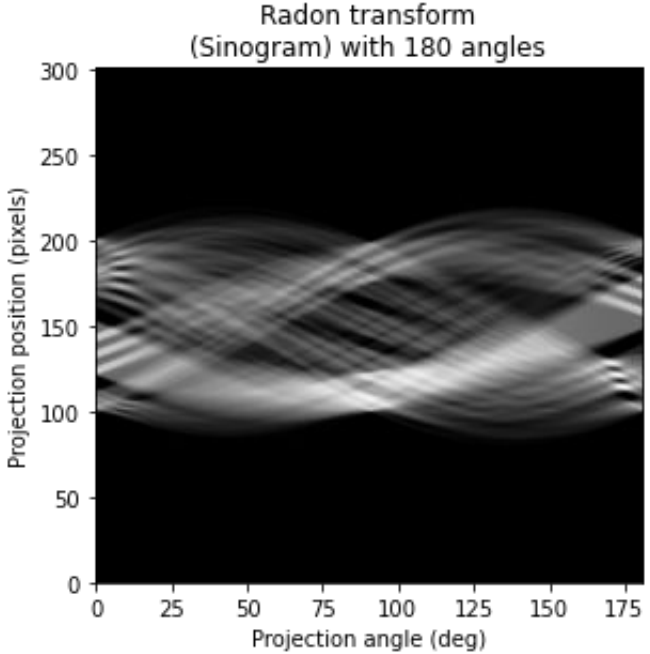




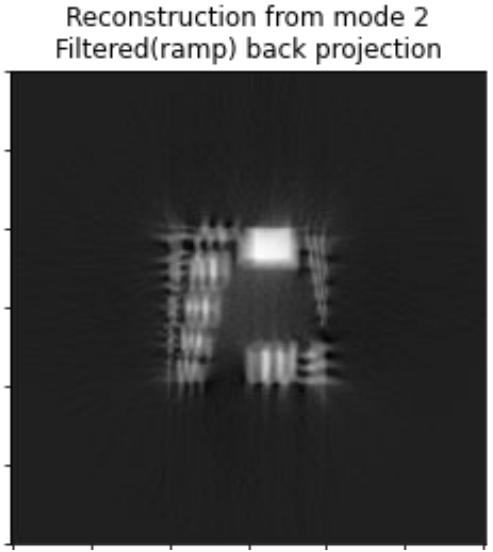


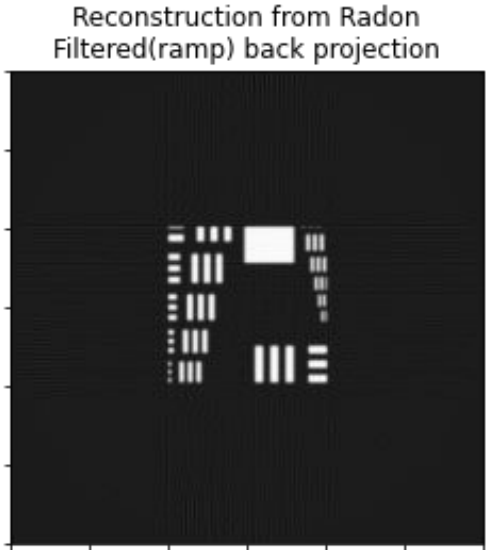
In simulation



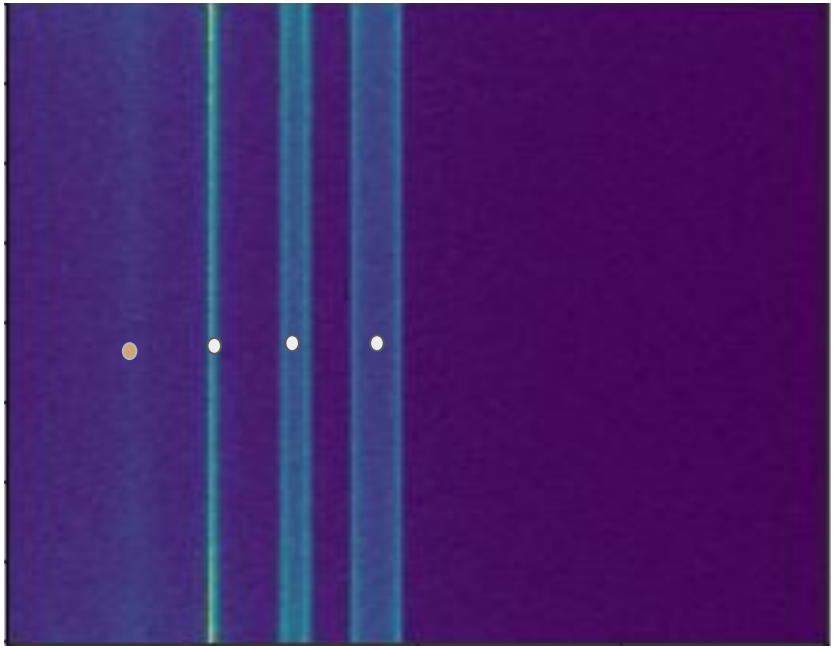


Reconstruction

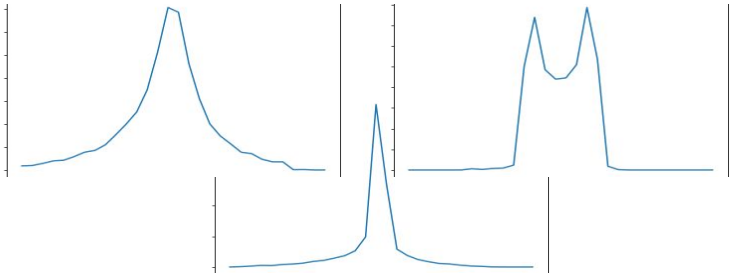




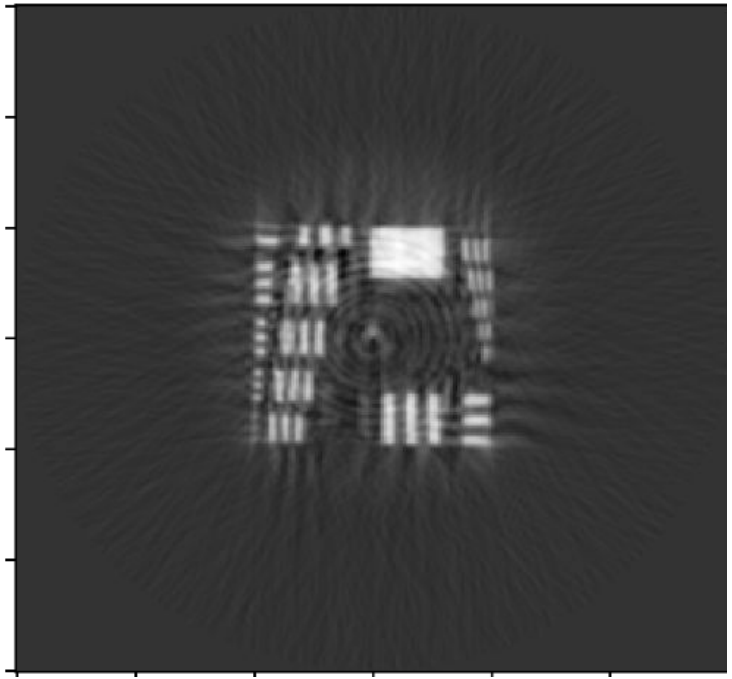
Improvement with a 1D deconvolution



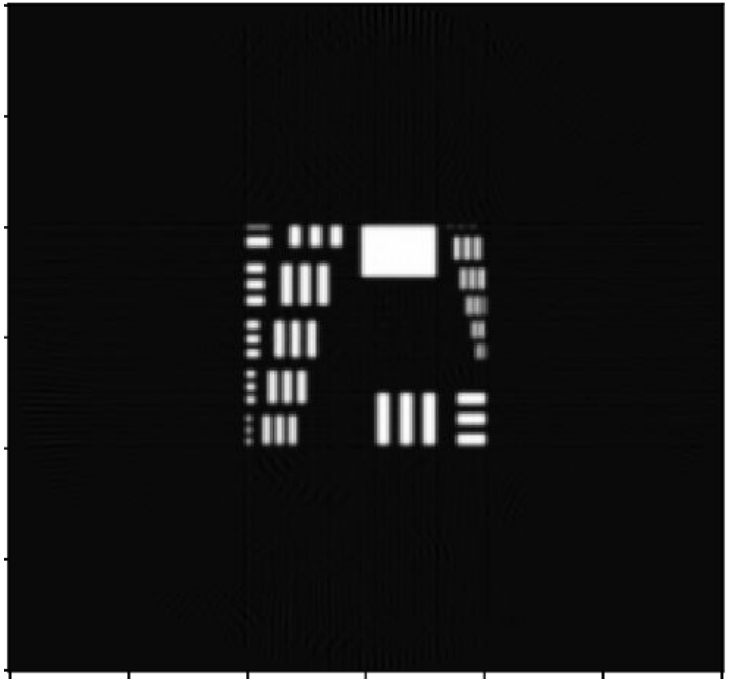
1D PSFs



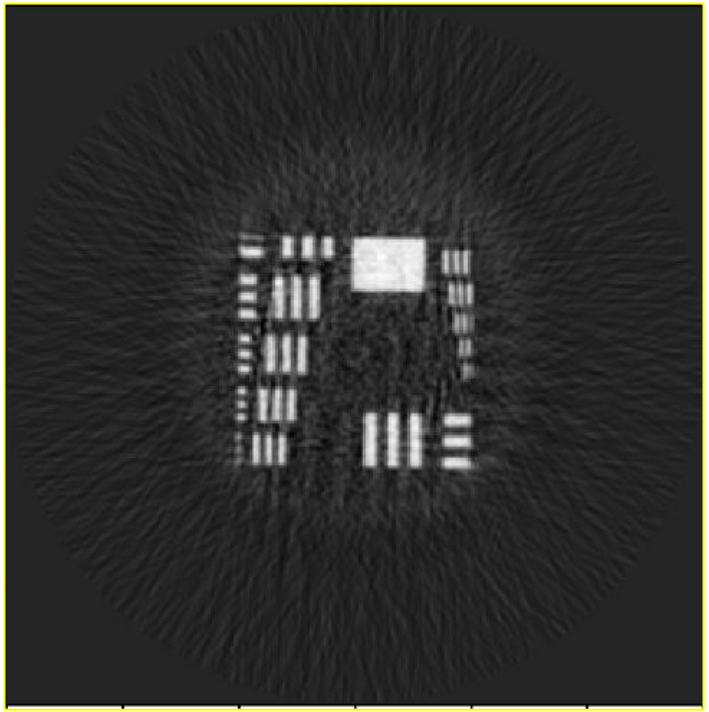
Sharpened mode 2 reconstructed



True image Radon and inverted

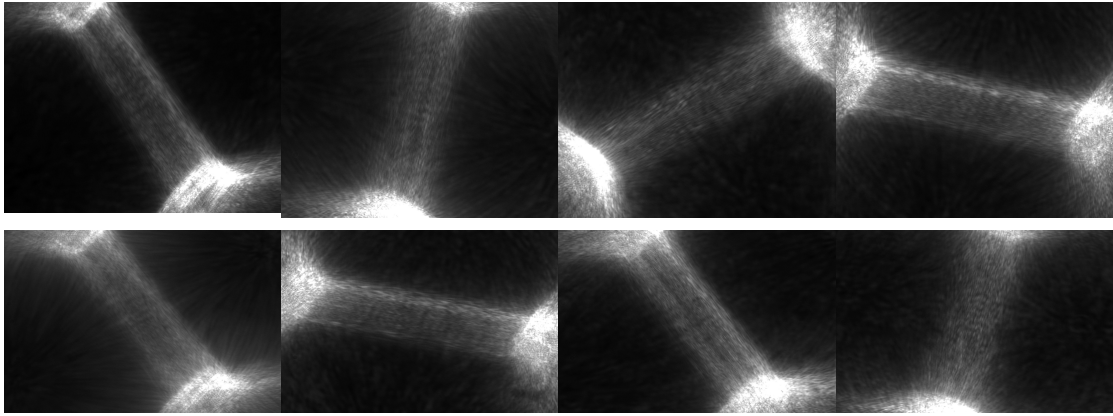


More improvement with more sets of images

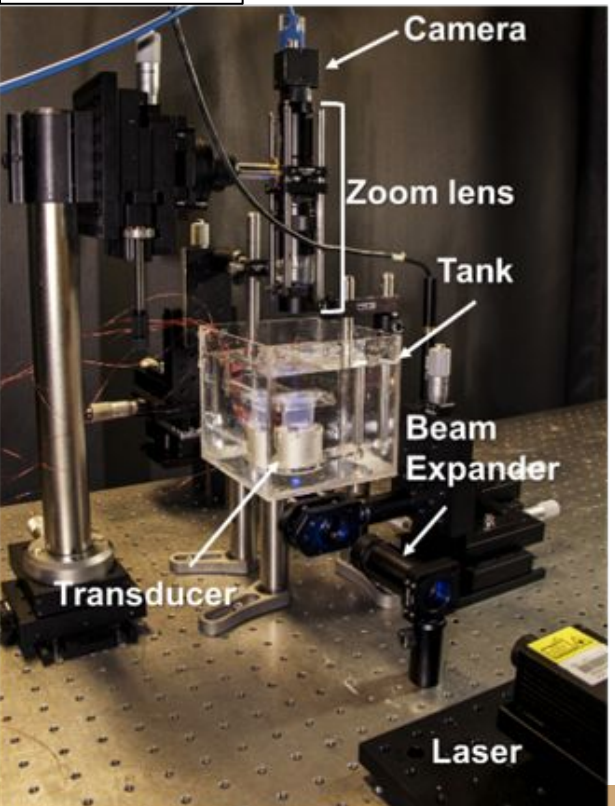


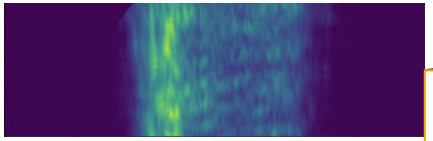
Captured images

Only 8 rotations for now (actually 4)

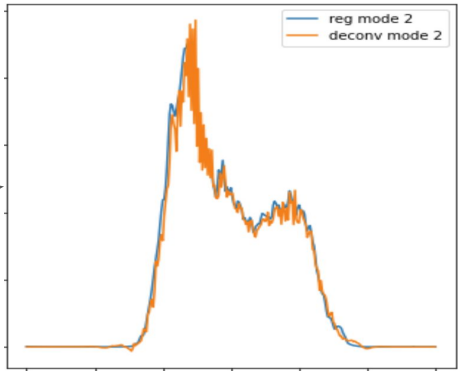


The setup:

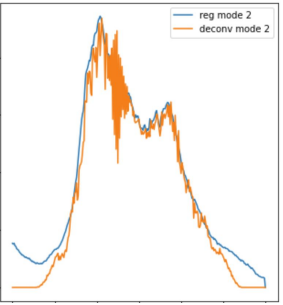


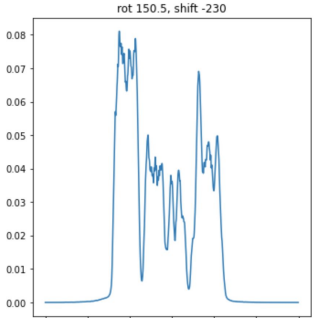


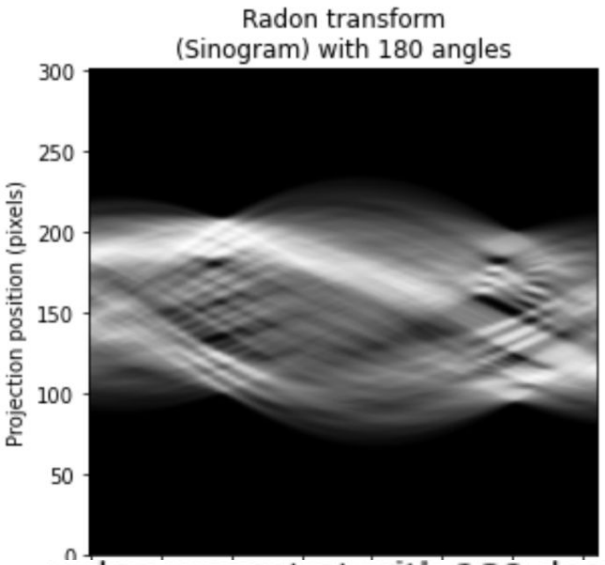
1D deconvolve

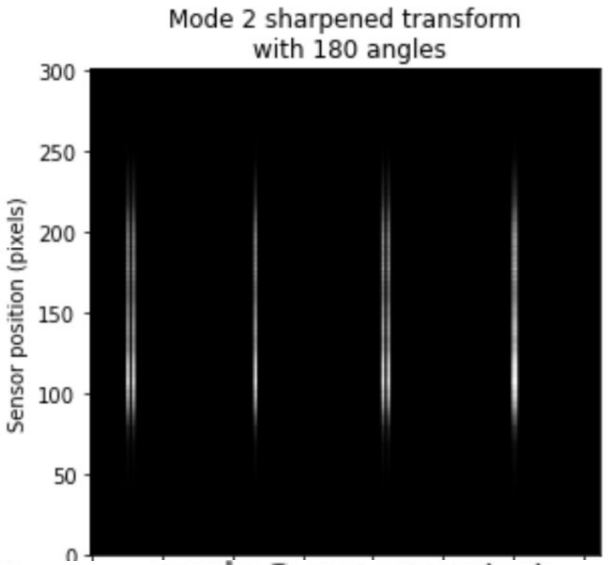


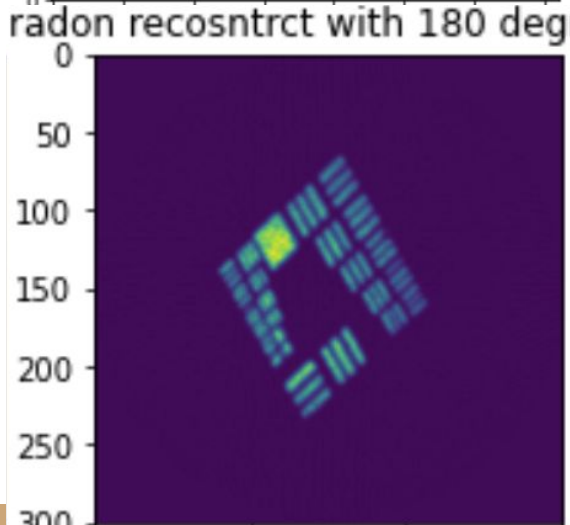
After 1D deconvolving

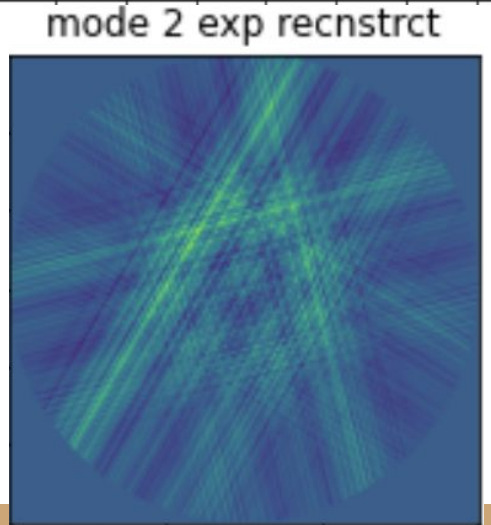




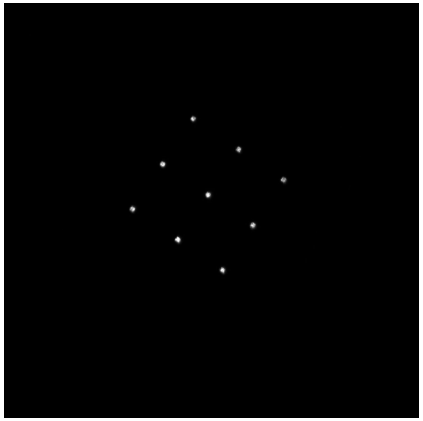


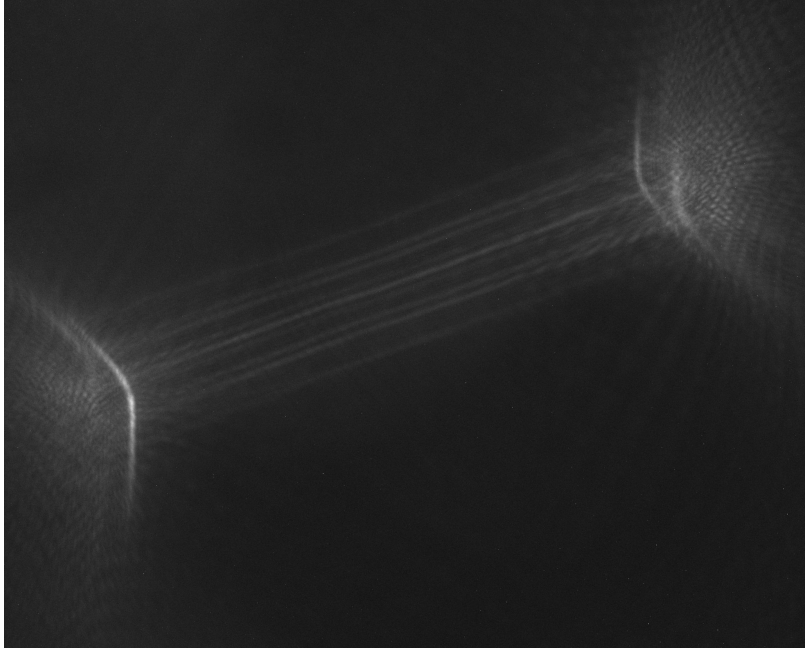




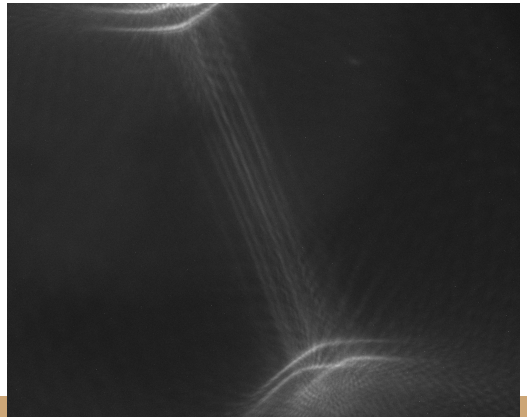


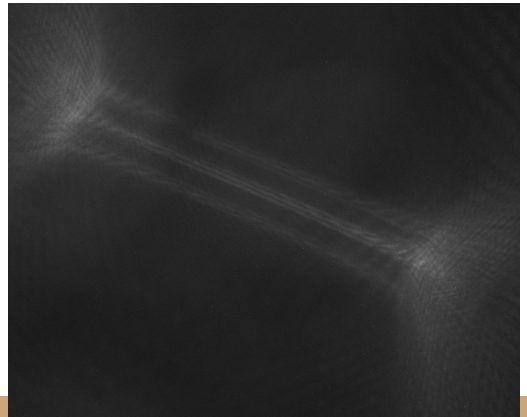
A sparse target image





More rotations:





Will it work?

- Yes :D
- Needs a better setup, to make more rotations possible

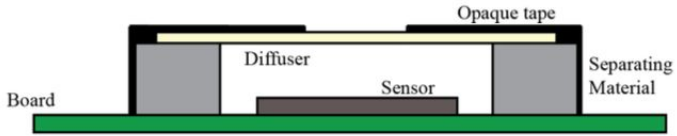
Thank You!

Building Diffuser Camera for Photography

Yi-Chun Chen

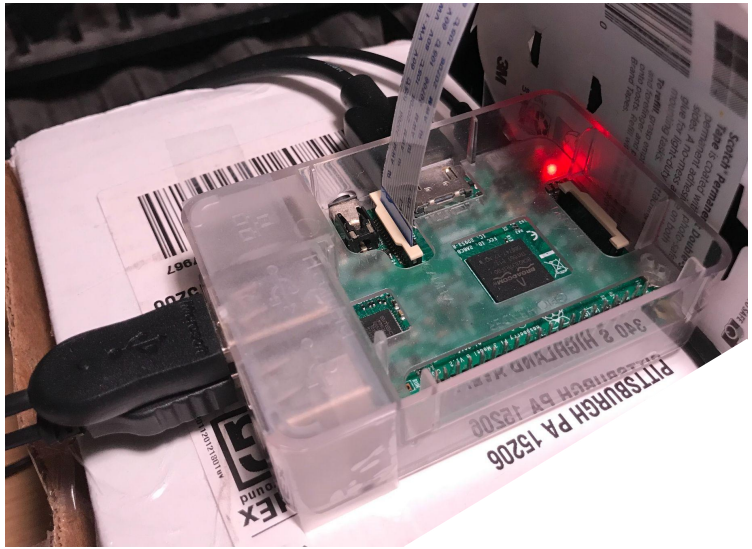
What is DiffuserCam?

- Lensless imaging system
  - Replace lens with a diffuser
- Advantages:
  - Lightweight
  - Create it at home!
  - Possibility of 3D imaging

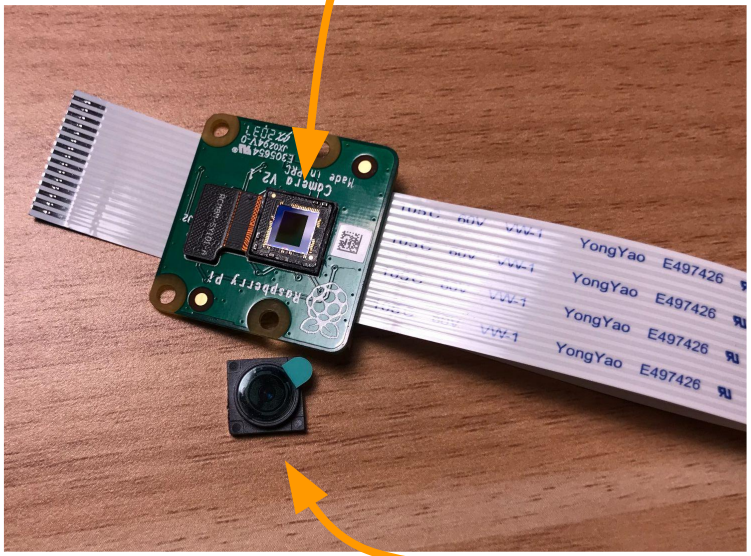


https://waller-lab.github.io/DiffuserCam/tutorial/building_guide.pdf

My DiffuserCam

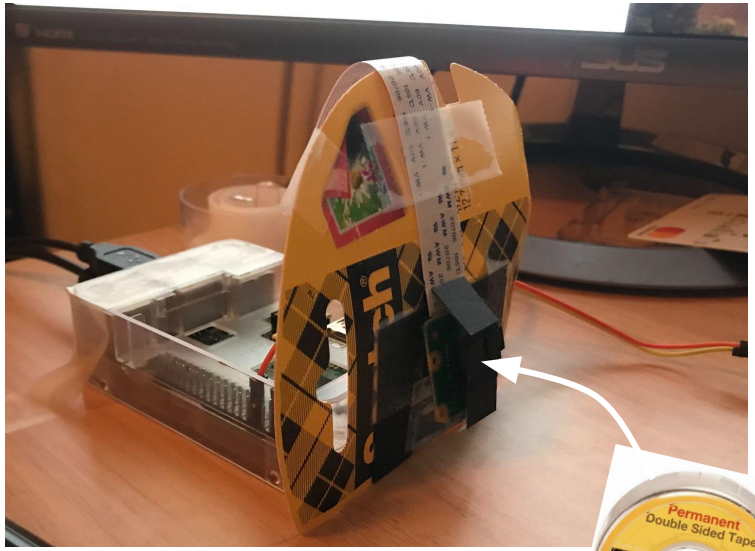


Raspberry Pi



Pi camera with lens removed

My DiffuserCam



Scotch Tape as the diffuser





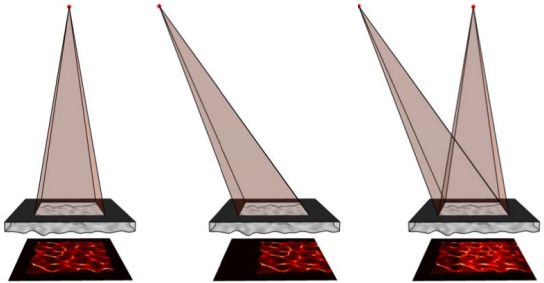
Black paper to construct aperture

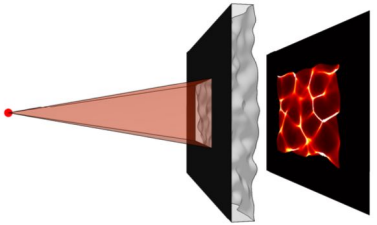
Why could it work?

$$f(v) = b = h * v$$

$\boldsymbol{v}$: a 2D array of light intensity values (scene),
the sum of many point sources of varying intensity and position
$\boldsymbol{b}$: a 2D array of pixel values on the sensor
$\boldsymbol{h}$: PSF

$$\text{crop}(h * v) \iff \mathbf{CMv}$$
$$f(v) \iff \mathbf{Av}$$





https://waller-lab.github.io/DiffuserCam/tutorial/building_guide.pdf

Why could it work?

$$f(v) = b \implies v = f^{-1}(b)$$

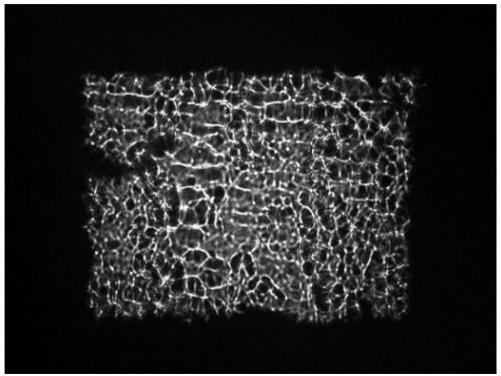
Solving a deconvolution problem.
Cropping function makes it not invertible.

$$\mathbf{v}^* = \underset{\mathbf{v}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{v} - \mathbf{b}\|_2^2$$

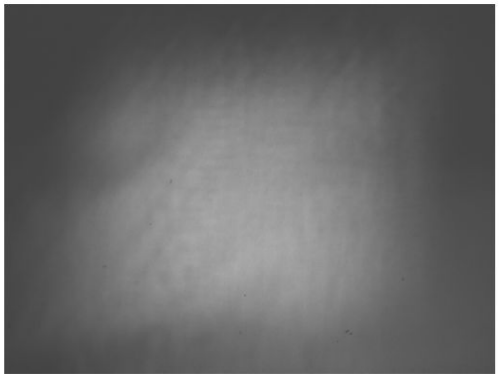
Use alternating direction method of multipliers (**ADMM**) to optimize the reconstruction.

https://waller-lab.github.io/DiffuserCam/tutorial/building_guide.pdf

Process



(a) Caustic image of a single point source



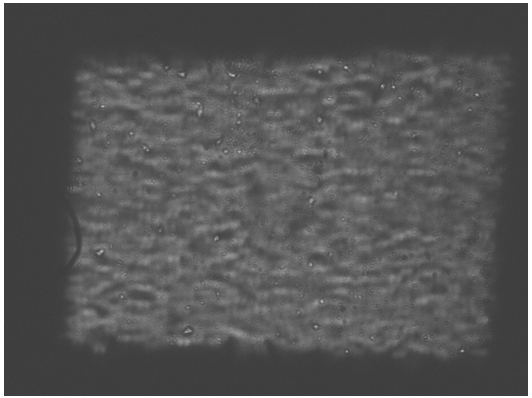
(b) Sensor reading of a hand



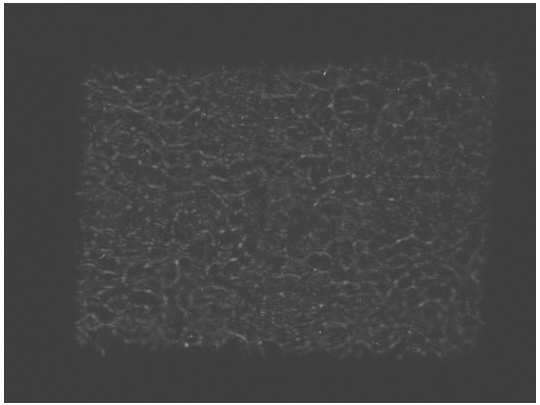
(c) Reconstructed image of a hand

https://waller-lab.github.io/DiffuserCam/tutorial/building_guide.pdf

Calibration and images



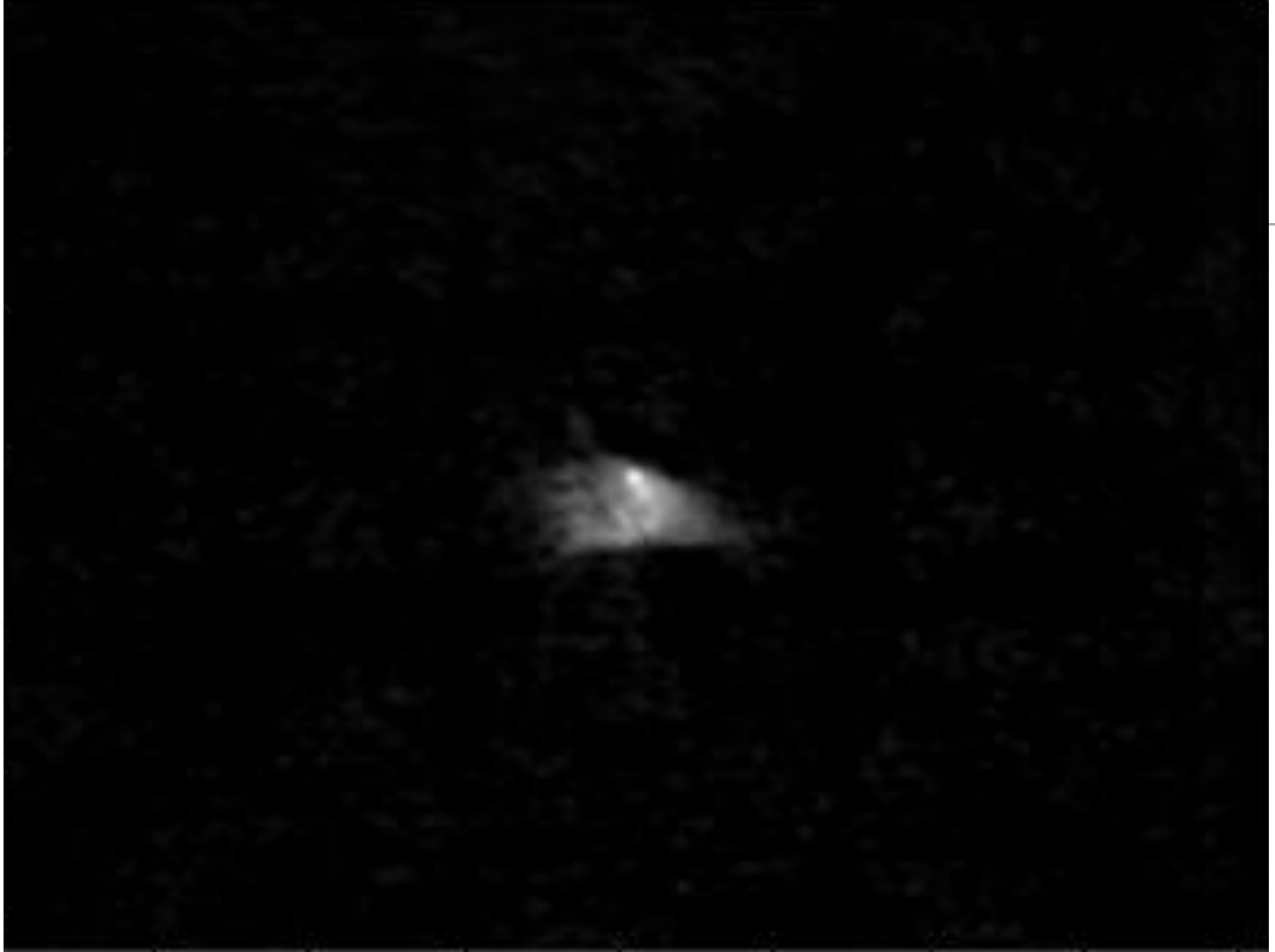
PSF - Out of focus

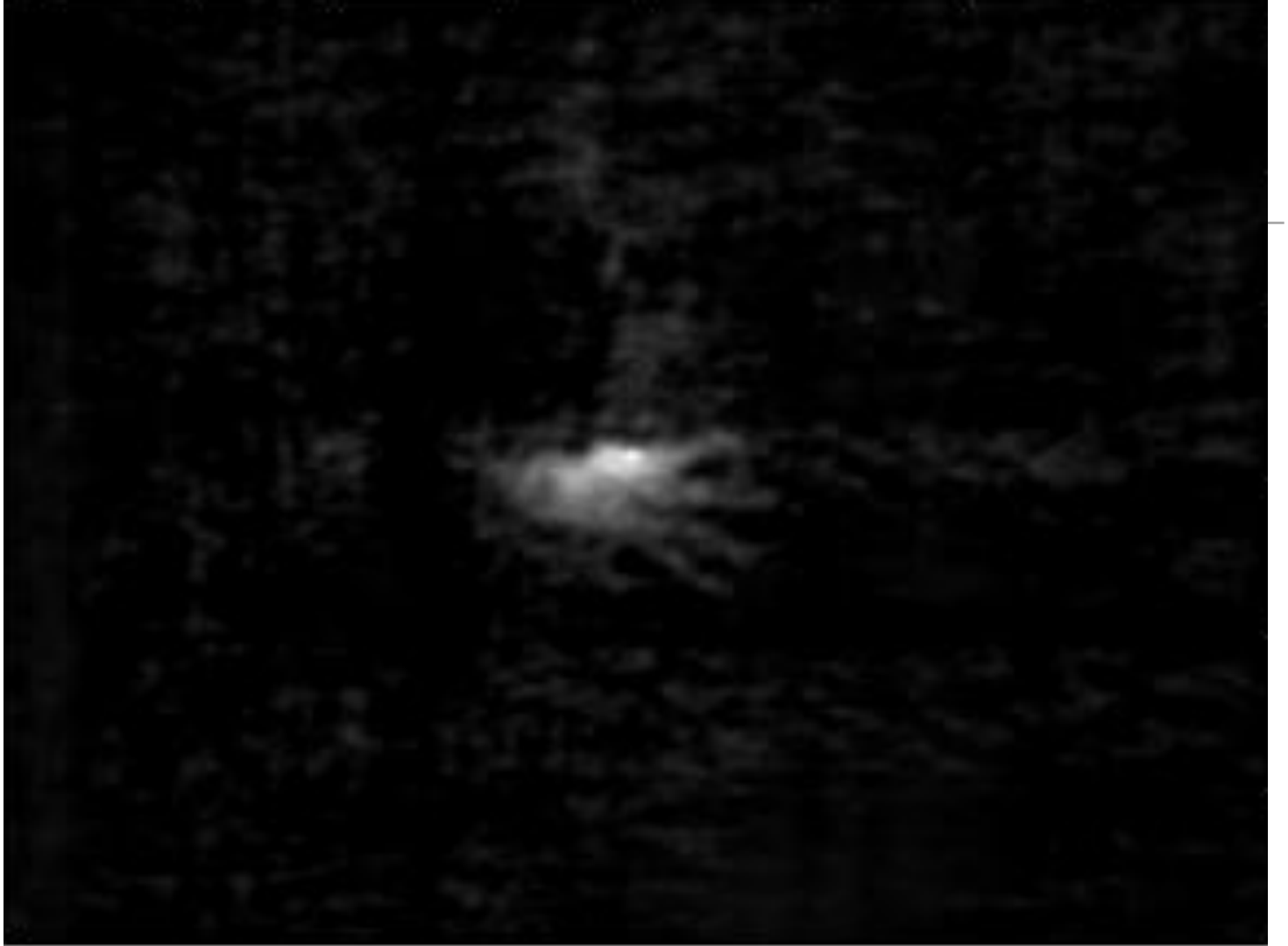


PSF - (Almost) in focus



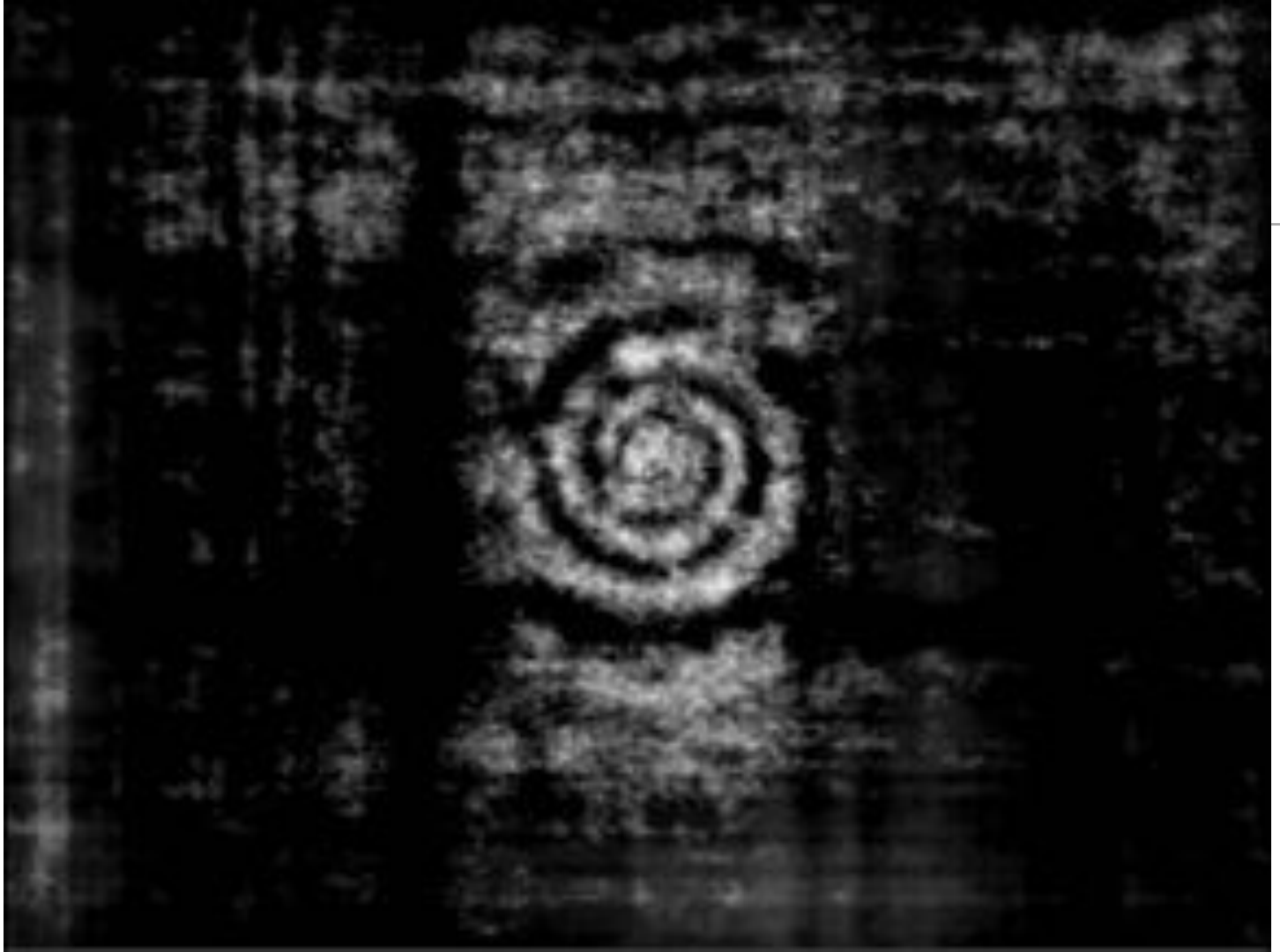
Sensor reading of an object





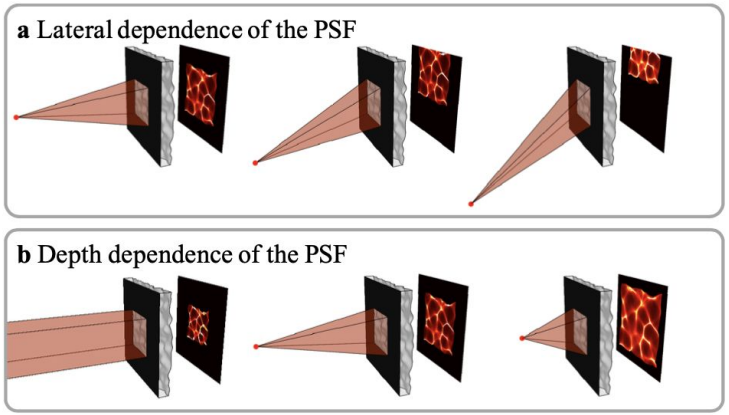


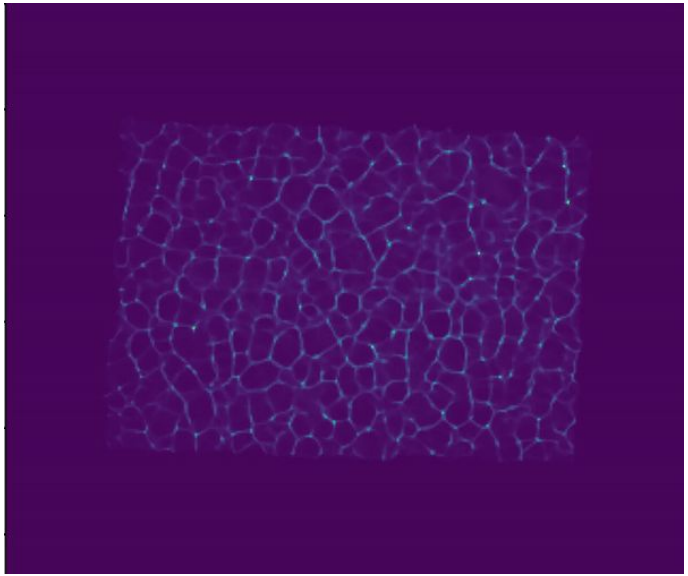


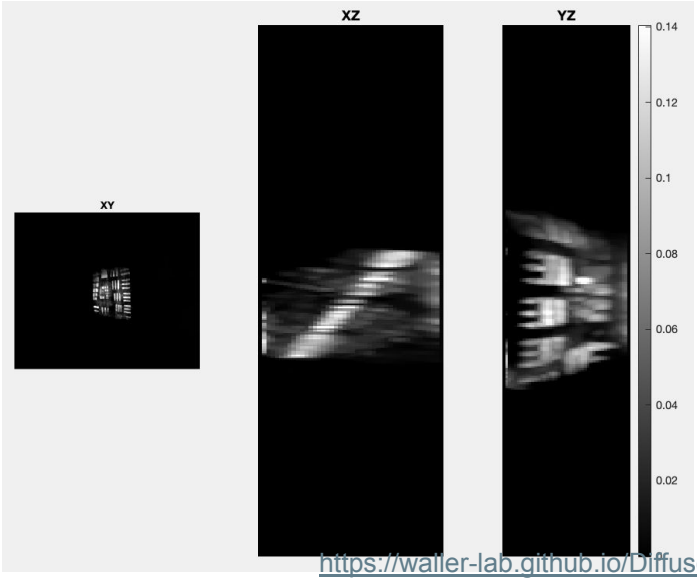


What's more

- 3D Imaging







https://waller-lab.github.io/DiffuserCam/

Acknowledgement

Ching-Yi Lin (ECE)
Wu-Chou Kuo (III)
Joon Jang
Prof. Yannis

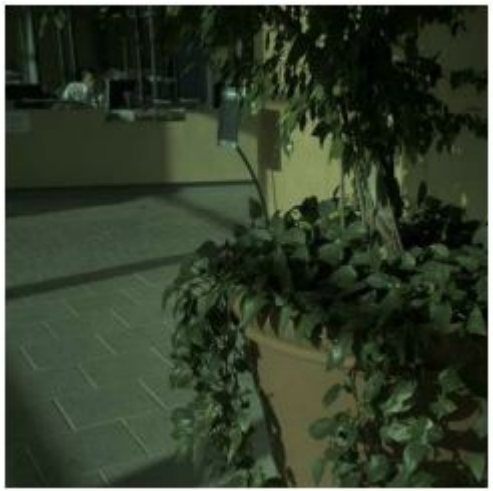
Convolution Color Constancy for Mobile Device Photography

Yutong Dai

Background

Color Constancy

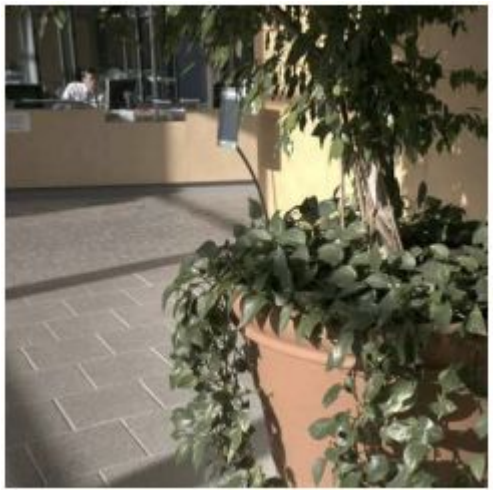
— What color is the light illuminating the scene?



[2]







[2]

Goals

Requirements for Color Constancy

— speed of processing, mitigation of uncertainty,
temporal cohesion for frames, account for low input resolution

Objective

— generalizability of introduced algorithm to photograph sets other than the ones utilized and curated from DSLR cameras



[1]

Algorithm: Convolutional Color Constancy (CCC)

Discriminative Learning

$$(\hat{L}_u, \hat{L}_v) = \arg\max_{u,v} (N * F)$$

— Optimizing:

$$\min_F \lambda \sum_{u,v} F(u,v)^2 + \sum_{i,u,v} P(u,v)\, C\left(u, v, L_u^{(i)}, L_v^{(i)}\right)$$

$$P(u,v) = \frac{\exp\left((N^{(i)} * F)(u,v)\right)}{\sum_{u',v'} \exp\left((N^{(i)} * F)(u',v')\right)}$$

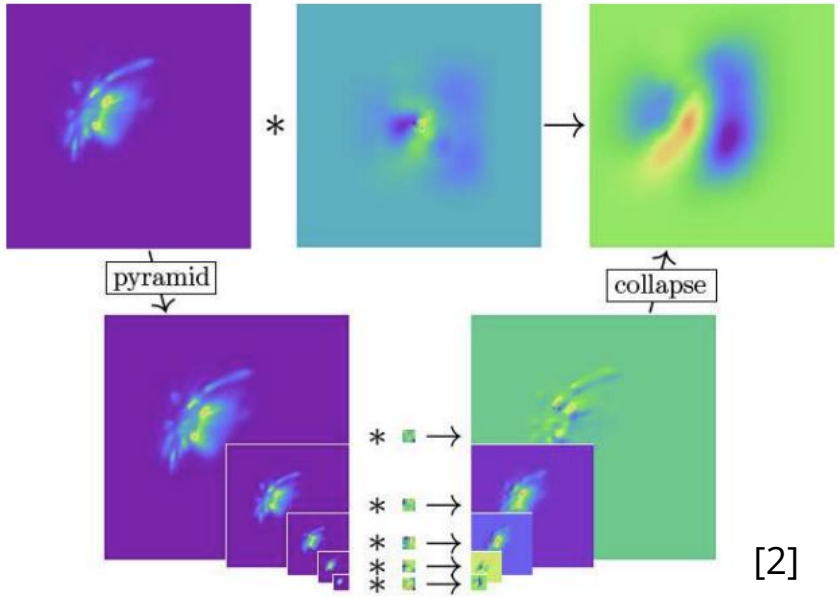


(a) Input Image (b) True Image (c) Tinted Image [2]

Algorithm: CCC

Efficient Filtering

naive
histogram
depiction



[2]

Algorithm: CCC

Generalization

— create more potential inputs with the same properties and histograms

$$
\begin{aligned}
I'_1 &= I \\
I'_2 &= \max\left(0, I * \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}\right) \\
I'_3 &= \mathrm{blur}(I^4, 11)^{1/4} \\
I'_4 &= \sqrt{\mathrm{blur}(I^2, 3) - \mathrm{blur}(I, 3)^2}
\end{aligned}
$$

Sample Results



— two sources of lighting



— only room lighting

Citations

1. Nikola Banic, Karlo Koscevic, Marko Subasic, and Sven Loncaric. The past and the present of the color checker dataset misuse. *CoRR*, abs/1903.04473, 2019
2. Jonathan T. Barron. Convolutional color constancy. *In Proceedings of the IEEE International Conference on Computer Vision (ICCV),* December 2015

Thank you!

Dual Photography

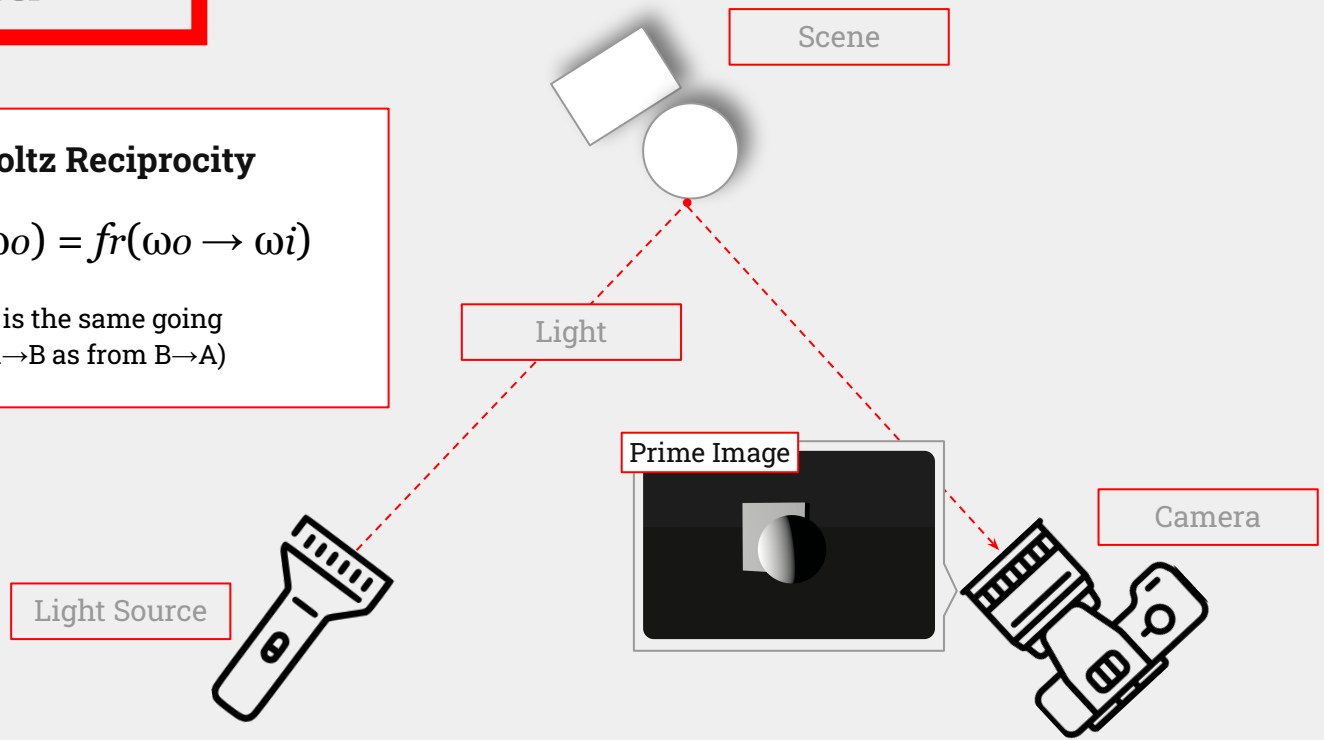
Ravi Dudhagra

Background

Helmholtz Reciprocity

$$fr(\omega i \rightarrow \omega o) = fr(\omega o \rightarrow \omega i)$$

(BRDF is the same going
from A→B as from B→A)

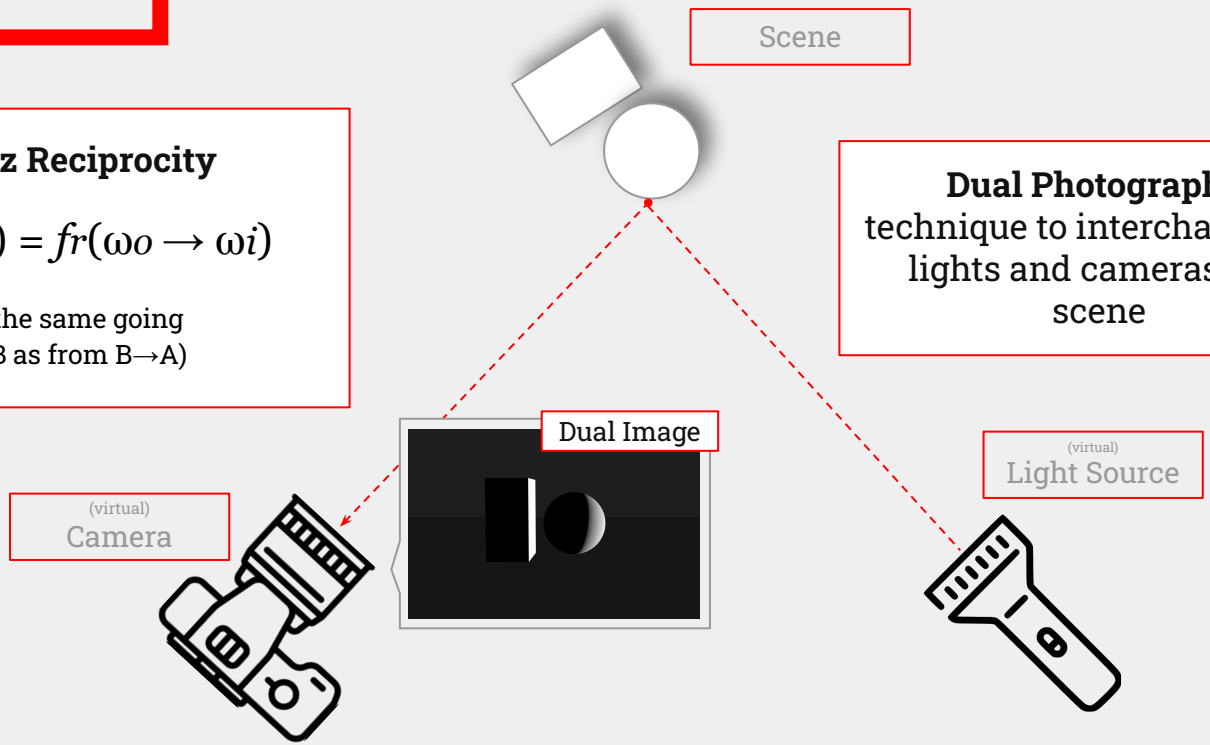


Background

Helmholtz Reciprocity

$$fr(\omega i \rightarrow \omega o) = fr(\omega o \rightarrow \omega i)$$

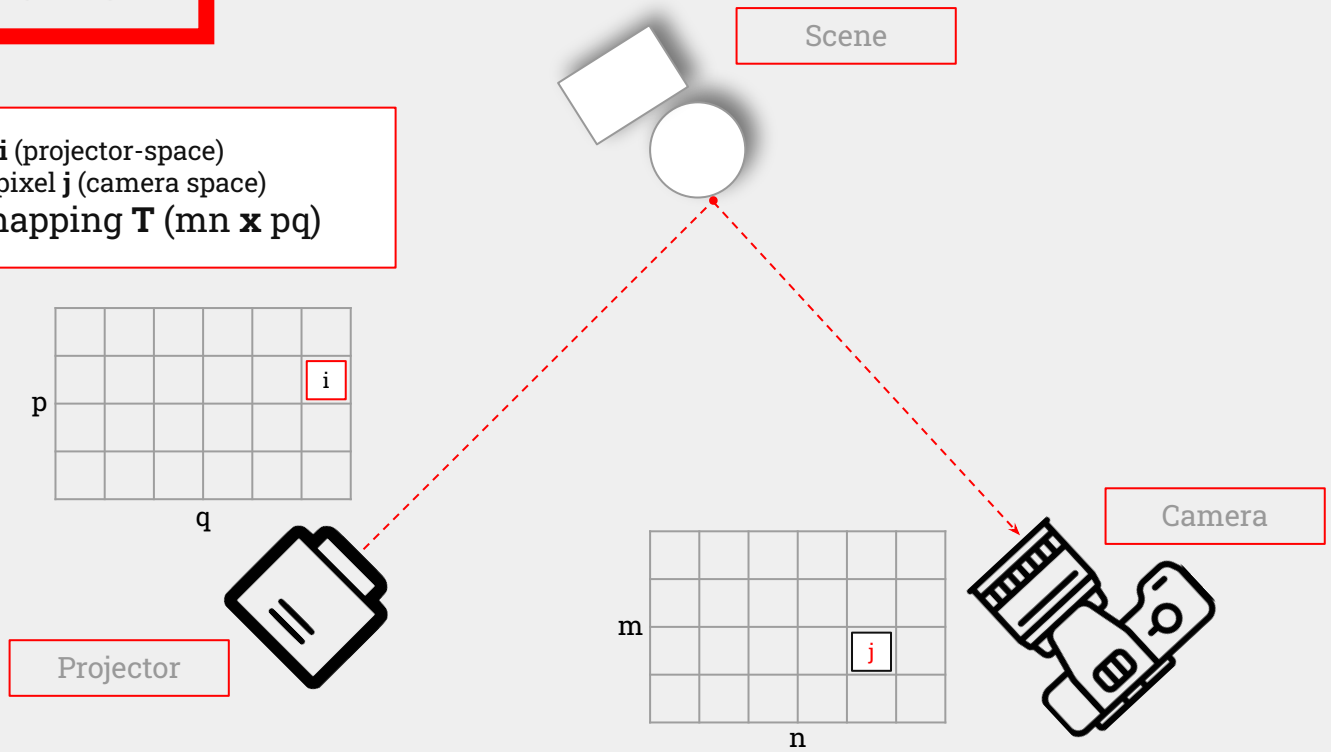
(BRDF is the same going from A→B as from B→A)



Dual Photography: technique to interchange the lights and cameras in a scene

Implementation

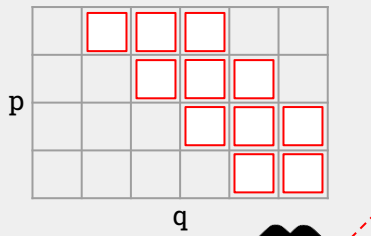
pixel **i** (projector-space)
maps to pixel **j** (camera space)
Call this mapping **T** (mn **x** pq)

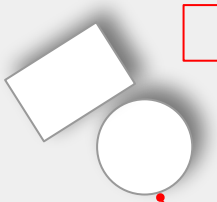


Implementation

c = T p

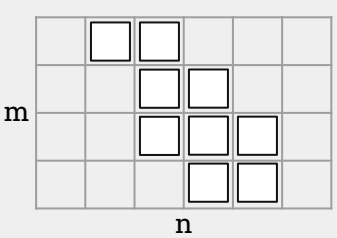
p is some pattern we project (pq x 1)
c is the resulting camera image (mn x 1)

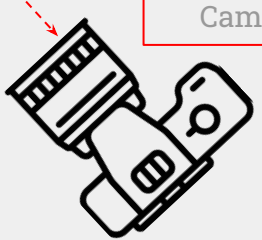




Scene

Projector



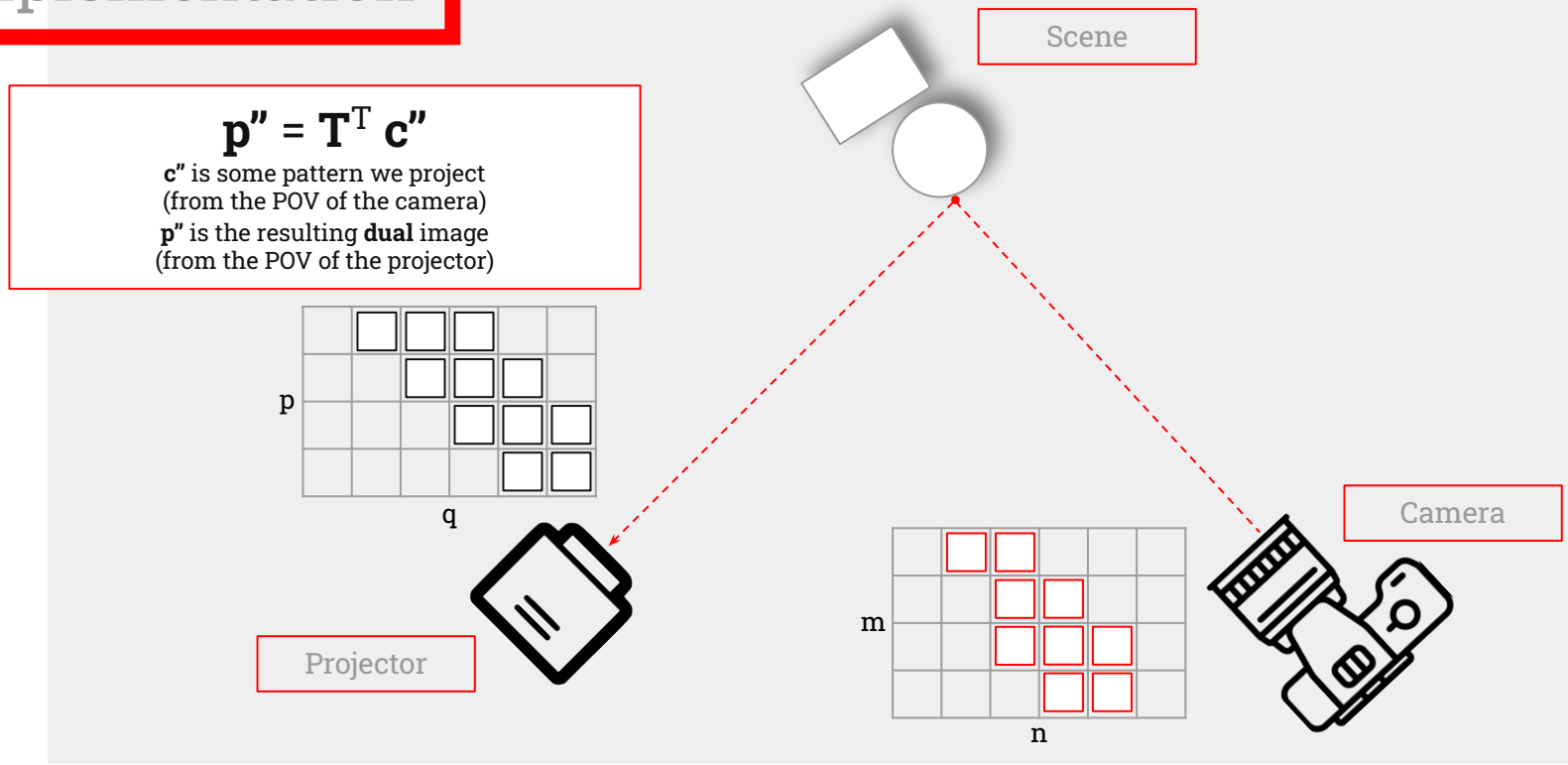


Camera

Implementation

$$p'' = T^T c''$$

c″ is some pattern we project
(from the POV of the camera)
p″ is the resulting **dual** image
(from the POV of the projector)



Acquiring **T**

(naive approach: capture an image for each projector pixel)

TWO PROBLEMS

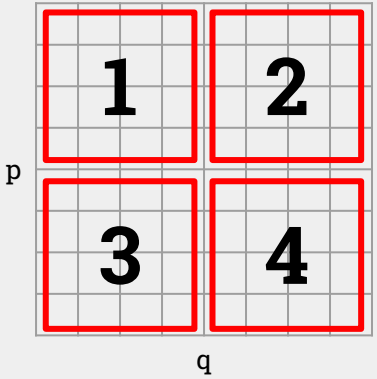
1) Too many projector pixels!
   - 1920*1080 = 2,073,600 images
2) **T** is VERY large
   - (1920*1080)*(1920*1080)*[8 bytes per num] = 34,398,535,680,000 (~34.4 TB)

Efficiently

Acquiring T

Use blocks

1) **Split projector into blocks**
2) Take image for each block
3) Identify blocks that didn't interfere with each other
4) Subdivide the blocks
5) Run non-interfering blocks in parallel

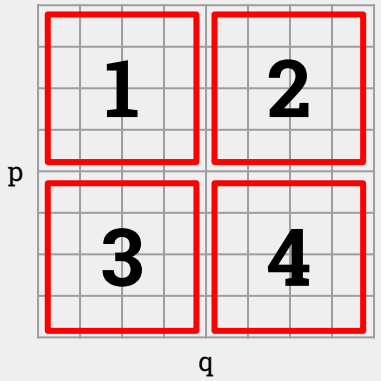


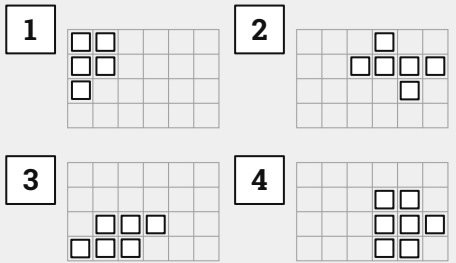
Efficiently

Acquiring T

Use blocks

1) Split projector into blocks
2) **Take image for each block**
3) Identify blocks that didn't interfere with each other
4) Subdivide the blocks
5) Run non-interfering blocks in parallel



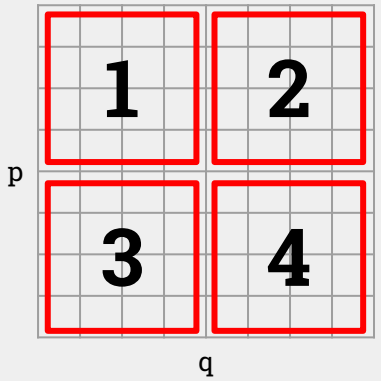


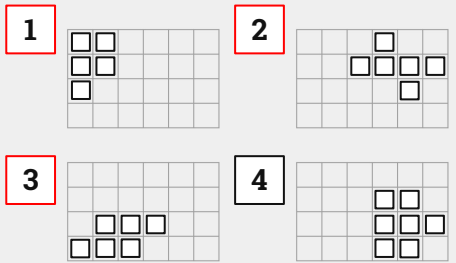
Efficiently

Acquiring T

Use blocks

1) Split projector into blocks
2) Take image for each block
3) **Identify blocks that didn't interfere with each other**
4) Subdivide the blocks
5) Run non-interfering blocks in parallel



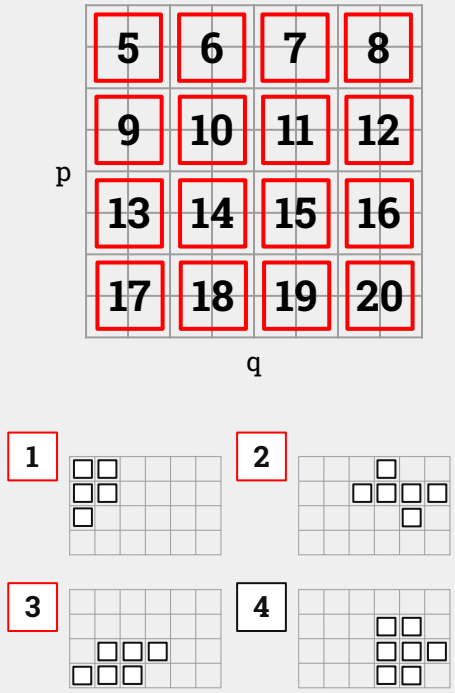


Efficiently

Acquiring T

Use blocks

1) Split projector into blocks
2) Take image for each block
3) Identify blocks that didn't interfere with each other
4) **Subdivide the blocks**
5) Run non-interfering blocks in parallel

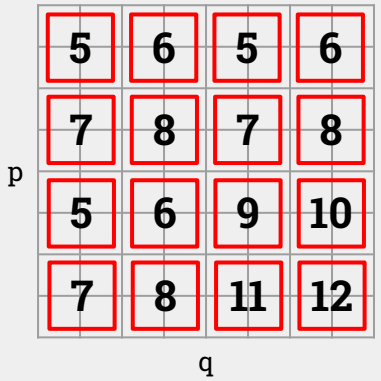


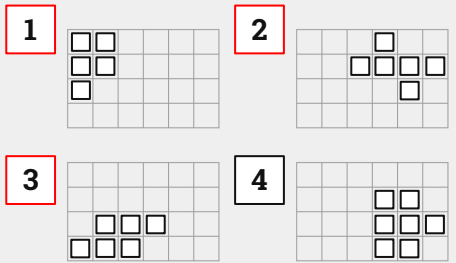
Efficiently

Acquiring T

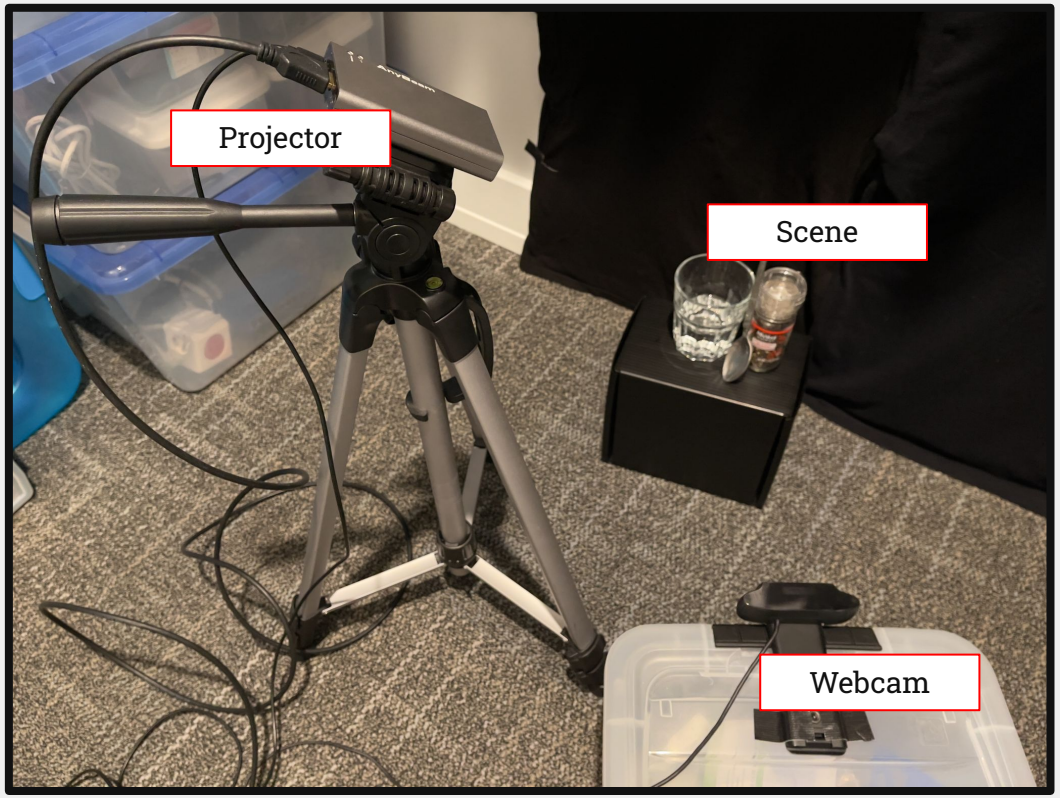
Use blocks

1) Split projector into blocks
2) Take image for each block
3) Identify blocks that didn't interfere with each other
4) Subdivide the blocks
5) **Run non-interfering blocks in parallel**





Setup



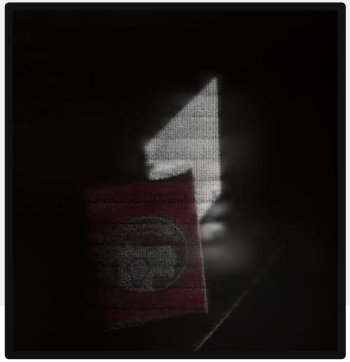
Results



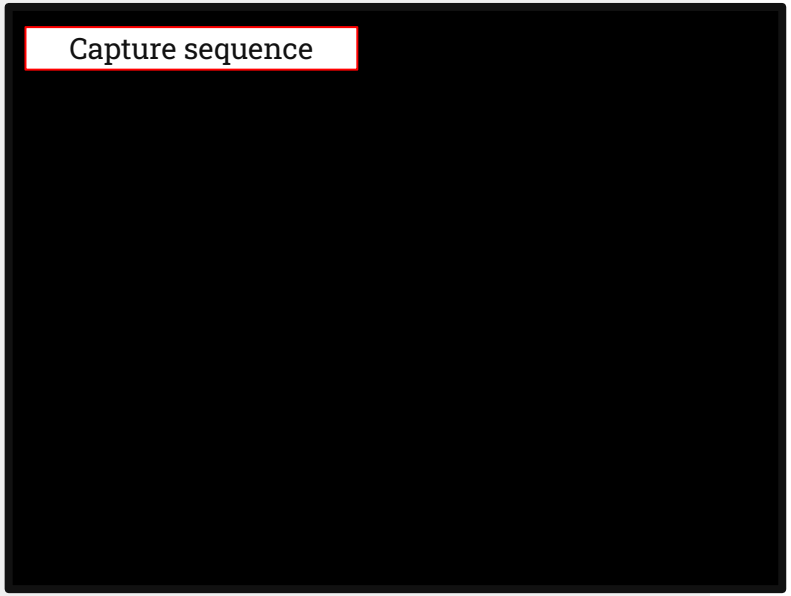












Results

Camera: (960 x 540)

Projector: (1280 x 720)

	# Subdivision levels	# Blocks	File size of T	Raw size of T matrix
Coffee Mug	16	943218	66MB	3.8TB
Box/cards	15	576833	37.7MB	3.8TB
Glass/spoon	15	159825	11.2MB	3.8TB

Applications



Capturing lightfields

- Multiple light sources cannot be used in parallel
- Solution: use multiple cameras instead, dual-photography to convert

thanks!

Any questions?

See the code @

https://github.com/rdudhagra/dual-photography

Read the paper @

https://graphics.stanford.edu/papers/dual_photography/



A Comparison of Various Edge-Aware Filtering Techniques

By Gilbert Fan

Bilateral Filter

Simple weighted average where weights are determined by similarity (sig_r) and distance (sig_s).

$$bf(I)_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in N(\mathbf{p})} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)\, I_{\mathbf{q}} \quad (1a)$$

$$W_{\mathbf{p}} = \sum_{\mathbf{q} \in N(\mathbf{p})} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) \quad (1b)$$

Domain Transform

Break image into horizontal and vertical 1D signals (C + 1 dimensions).

Domain transform into 1D, must be isometric.

Apply a 1D filter to the transformed data repeatedly for set number of iterations.

Guided Filter

“Similar” to joint bilateral filter but instead of computing weights from guide image, the output is a linear transformation of the guide image with coefficients determined by the input image, calculated window by window.

OutputWindow = a * GuideWindow + b

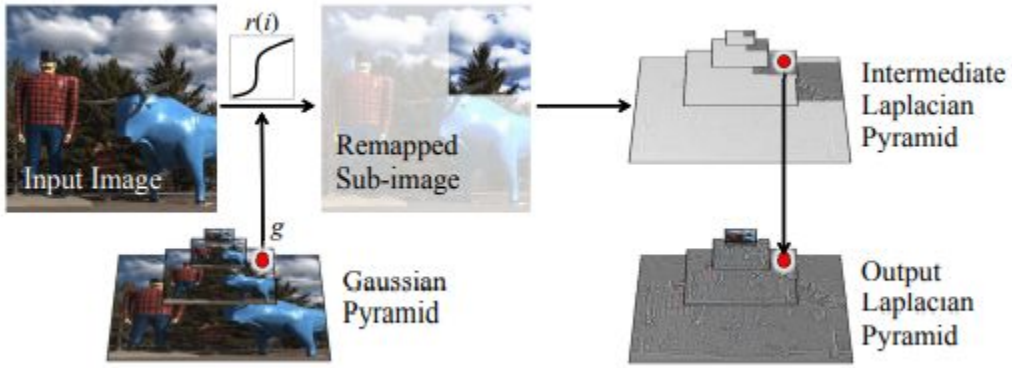
Where a, b are determined by the input image window

High variance around pixel => a = 1, b = 0

Flat patch => a = 0, b = average of window

Laplacian Filter

Constructs a laplacian pyramid pixel by pixel from remappings of the input image using values of its gaussian pyramid, collapsing them to form the final image.



Bilateral



Domain



Guided



Laplacian



Original



Bilateral



Domain



Guided



Laplacian



Original



Bilateral



Domain



Domain





Guided



Laplacian



Citations

CHEN, J., PARIS, S., AND DURAND, F. 2007. Real-time edge-aware image processing with the bilateral grid. ACM Transactions on Graphics (Proc. SIGGRAPH) 26, 3.

Eduardo S. L. Gastal and Manuel M. Oliveira. "Domain Transform for Edge-Aware Image and Video Processing". ACM Transactions on Graphics. Volume 30 (2011), Number 4, Proceedings of SIGGRAPH 2011, Article 69.

He, K., Sun, J., & Tang, X. (2013). Guided Image Filtering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(6), 1397–1409. https://doi.org/10.1109/tpami.2012.213

Paris, S., Hasinoff, S. W., & Kautz, J. (2015). Local laplacian filters. Communications of the ACM, 58(3), 81–91. https://doi.org/10.1145/2723694





2D Light Direction Estimation Analysis Using Different Color Spaces

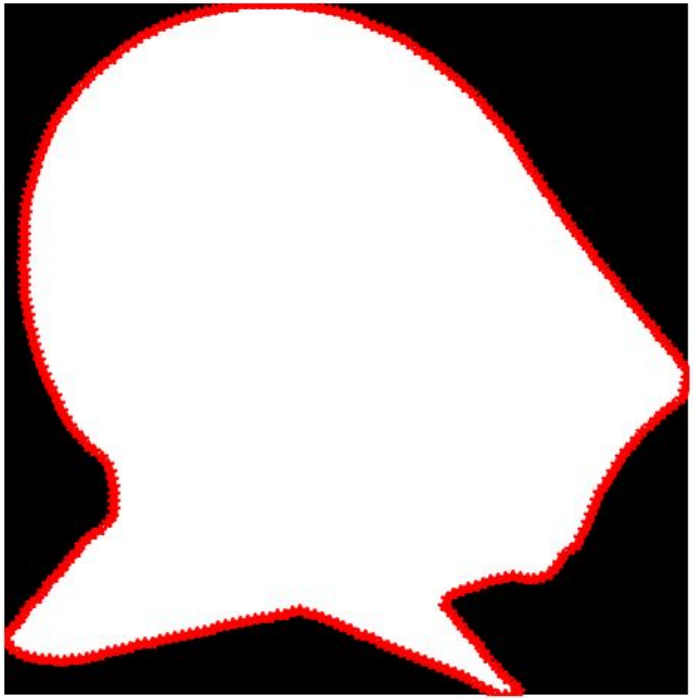
Mary Hatfalvi

2D Light Direction Estimation

Input Image



Contours traced around image mask



2D Light Source Estimation Algorithm [1]

Least Squares Solution for Light Direction [1]

$$E(\vec{L}, A) = \left\| M \begin{pmatrix} L_x \\ L_y \\ A \end{pmatrix} - \begin{pmatrix} I(x_1, y_1) \\ I(x_2, y_2) \\ \vdots \\ I(x_p, y_p) \end{pmatrix} \right\|^2 = \left\| M\vec{v} - \vec{b} \right\|^2$$

This least squares equation can be rewritten as a pseudo inverse solution

L = Light Direction

N = 2D Normal Vectors (found from 2 connecting contour points)

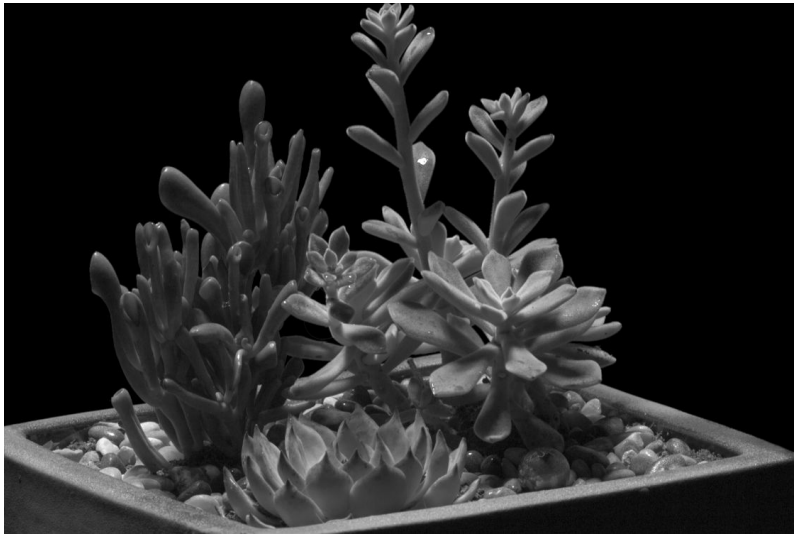
I = Intensity Value of the Light interpolated at the location of the normal vector

$$M = \begin{pmatrix} N_x(x_1, y_1) & N_y(x_1, y_1) & 1 \\ N_x(x_2, y_2) & N_y(x_2, y_2) & 1 \\ \vdots & \vdots & \vdots \\ N_x(x_p, y_p) & N_y(x_p, y_p) & 1 \end{pmatrix}$$

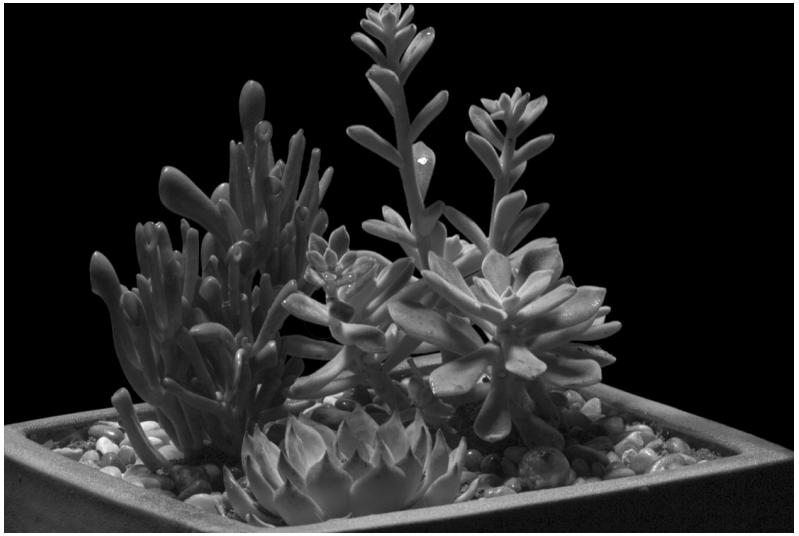
M Matrix Definition [1]

[1] Kee, Eric & Farid, Hany. (2010). Exposing digital forgeries from 3-D lighting environments. 10.1109/WIFS.2010.5711437.

Color Spaces

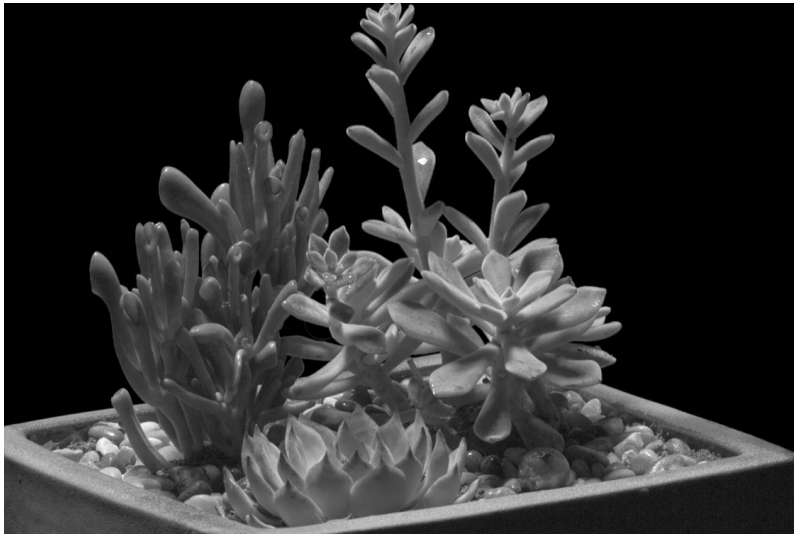


Grayscale

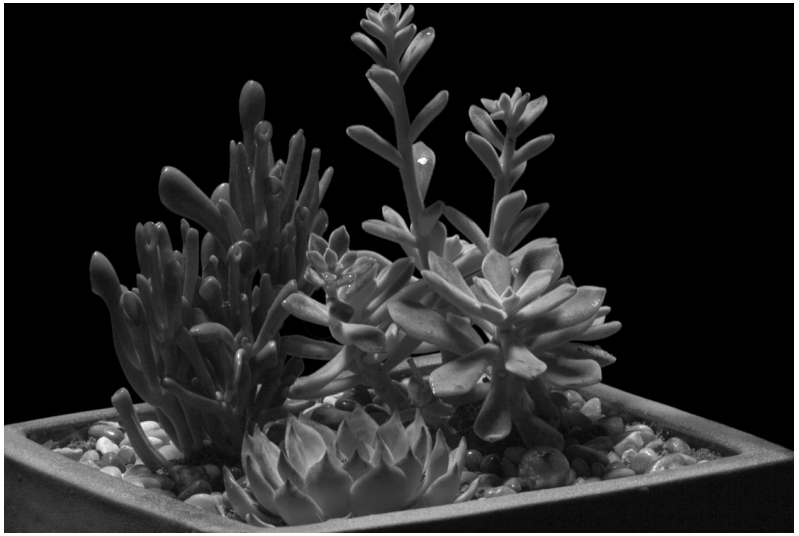


XYZ

Color Spaces

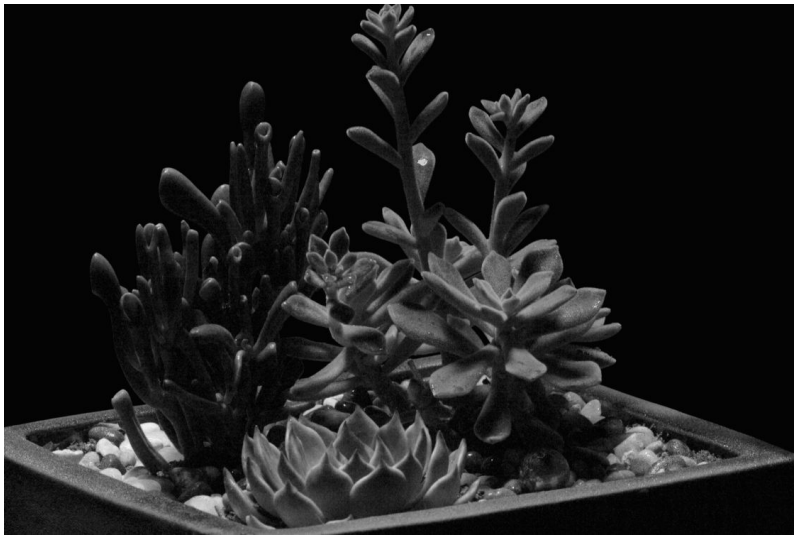


LAB

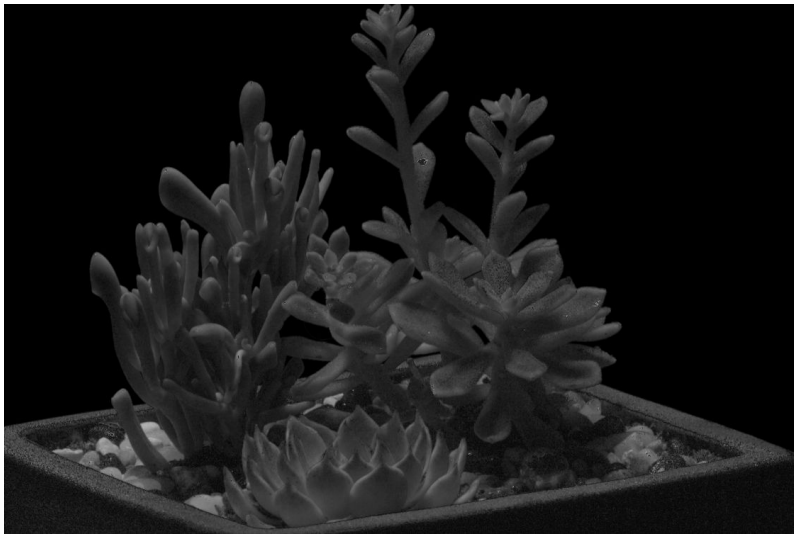


YCbCr

Color Spaces



Illumination Invariant - (IIv) [1]



SUV [2]

[1] Hamilton Y. Chong, Steven J. Gortler, and Todd Zickler. 2008. A perception-based color space for illumination-invariant image processing. In ACM SIGGRAPH 2008 papers (SIGGRAPH '08)DOI:https://doi.org/10.1145/1399504.1360660

[2] Satya P. Mallick, Todd E. Zickler, David J. Kriegman, and Peter N. Belhumeur, "Beyond Lambert: Reconstructing Specular Surfaces Using Color." in Proc. IEEE Conf. Computer Vision and Pattern Recognition, June 2005.

Dataset used for Testing



Helmet Left [1]



Plant Left [1]

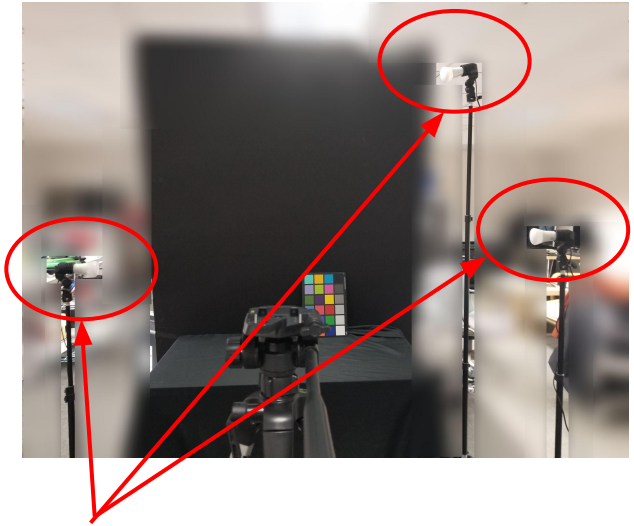
[1] “Light Stage Data Gallery.” *Light Stage Data Gallery*, https://vgl.ict.usc.edu/Data/LightStage/.

Helmet and Plant Error Results

Plant Left	Grayscale	XYZ	LAB	YCbCr	SUV	IIV
Mean Angle Error	0.7646	0.7188	0.7778	0.7677	0.5044	0.6431
Mean Euclidean Distance Error	0.4409	0.3995	0.455	0.445	0.1786	0.3064

Helmet Left	Grayscale	XYZ	LAB	YCbCr	SUV	IIV
Mean Angle Error	0.3939	0.6117	0.3701	0.3974	0.473	0.4593
Mean Euclidean Distance Error	0.1437	0.3434	0.1239	0.146	0.2227	0.1913

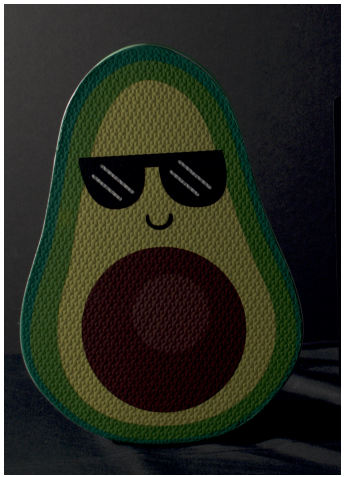
Taken Images



3 Light Sources



Example Image with Color Checker and Chrome Sphere for calculating 2D Light Direction [1]

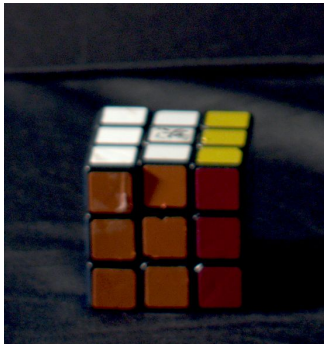




[1] Ying Xiong (2021). PSBox (https://www.mathworks.com/matlabcentral/fileexchange/45250-psbox), MATLAB Central File Exchange. Retrieved December 8, 2021.

Interesting Images Results (Rubix Cube)

Input Image



GT 2D Light Estimation

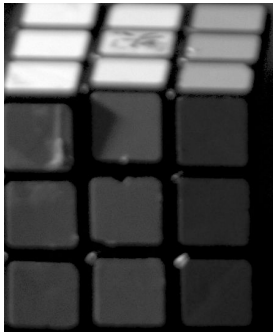




SUV - Specularity Invariant Combination



SUV Light Direction Estimation Results



XYZ - Y Channel



XYZ light Direction Estimation Results

Thank you

Dual Pixel Defocus Disparity:

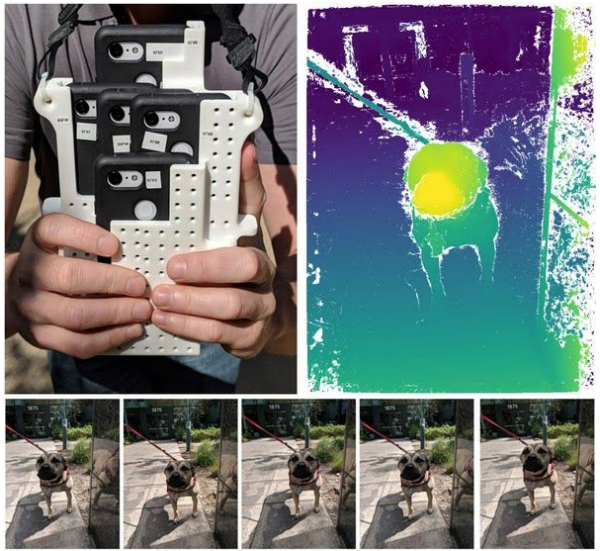
Depth Estimation using the Symmetric Point Spread Function Model

Anne He

Motivation

- Getting the depth map is very useful - refocusing gives nice stylistic effects, especially on lower end hardware
- How can we get an accurate depth map without needing to collect a ton of data?
  - Other methods either require multiple images or very constrained scenarios
  - Learning-based approaches require large amounts of ground truth training data which is hard to acquire
- Back to depth from defocus

??

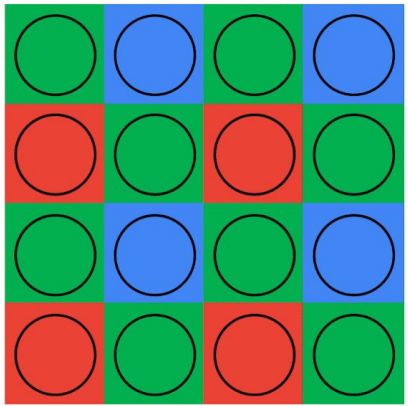


Garg et al. Learning Single Camera Depth Estimation using Dual Pixels.

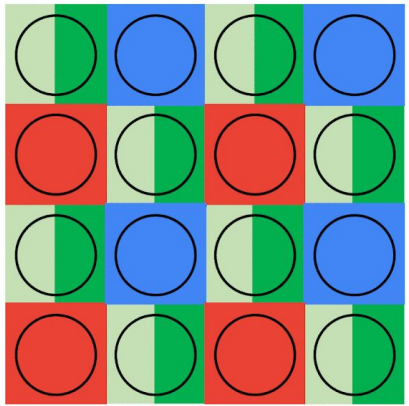
??

Dual Pixel (DP) Sensors

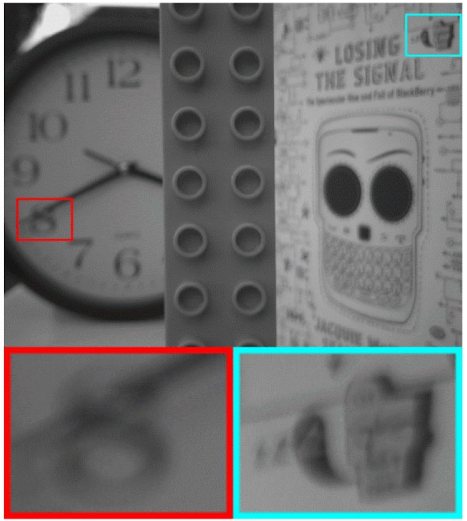
Left photodiode ⟵ ⟶ Right photodiode



Regular sensor



DP sensor



DP image



Canon EOS R5



Google Pixel 4

Smartphone DP data in the wild

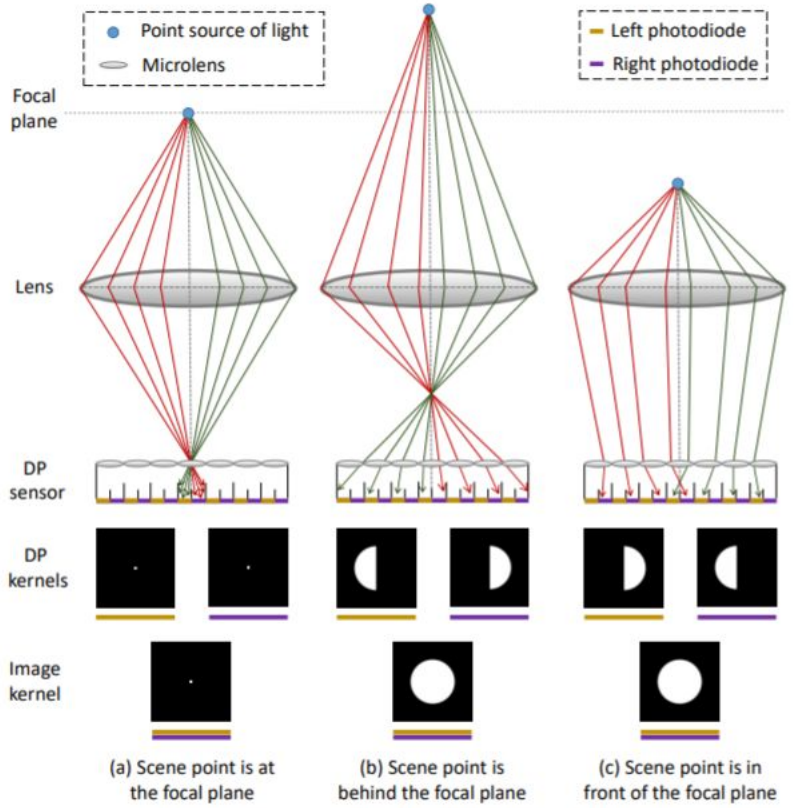


Smaller aperture leads to smaller disparity, but still noticeable to the human eye

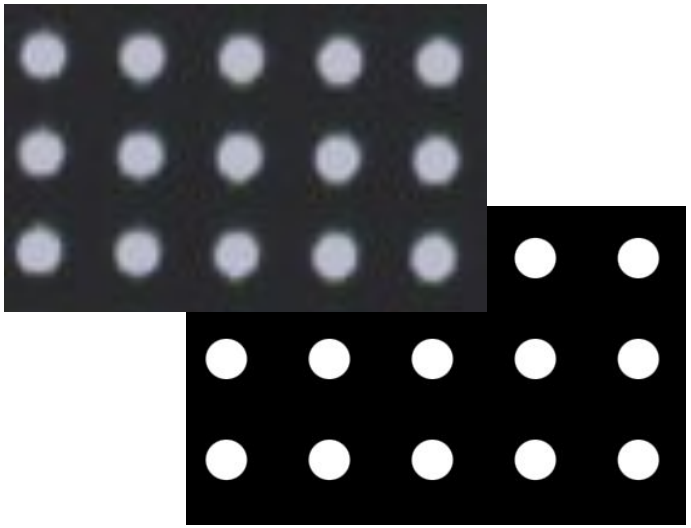


Hypothesis

- In an ideal DP camera, the circle of confusion is evenly split between the two views and sum to a full circle
- In real life, light may leak into the opposite side
- How do we prove this property still generally holds?



Experiment



- Capturing the PSF
  - The PSF models what a point light source looks like under different depths and camera configurations
  - Can also estimate PSF using image of grid of disks with known radius and spacing (the sharp image)
- Consider a constant depth image patch G and the known sharp image F
- Let H be the PSF that produces G when convolved with F

$$G_l = F * H_l$$

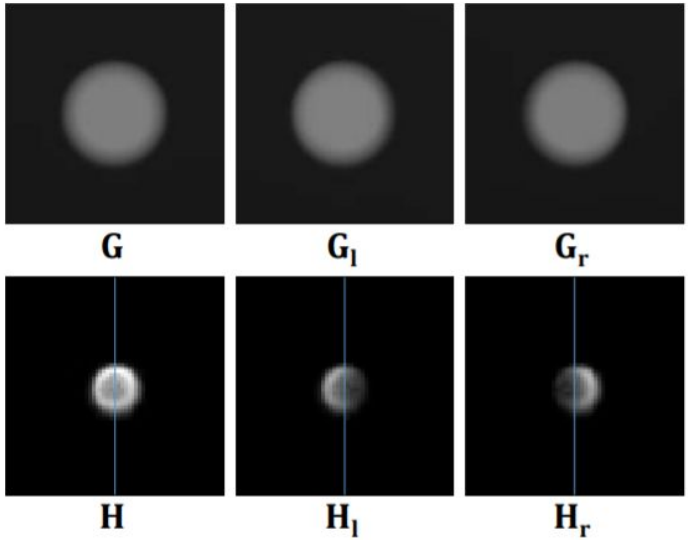
$$G_r = F * H_r$$

$$G = G_l + G_r = F * (H_l + H_r) = F * H$$

Are kernels symmetrical?

- Estimate kernel for each view
- Cross-correlate two sides

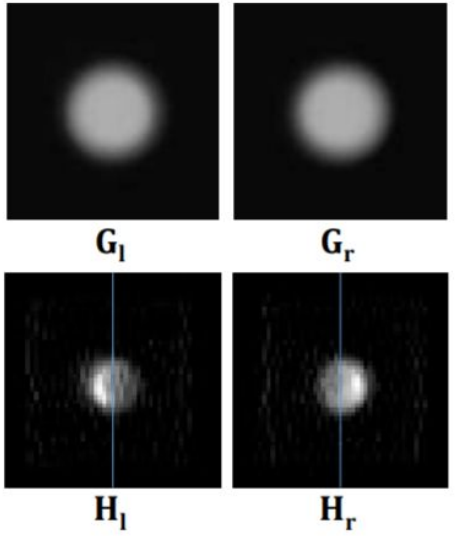
$$\underset{K \geq 0}{\arg\min} \sum_{p} ||D_p(F * K) - B||_2^2 + \gamma||K||_1$$



$\chi(H_l, H_r^f)_{DSLR} > 0.95$

$\chi(H_l, H_r^f)_{Pixel} > 0.8$

(mostly) Yes!



How is this useful for depth estimation

- Radius of blur kernel is in relation to depth!
  - Thin lens model: distance between sensor and lens creates circular blur
- Symmetry allows us to simplify parameters and solve an optimization problem for H

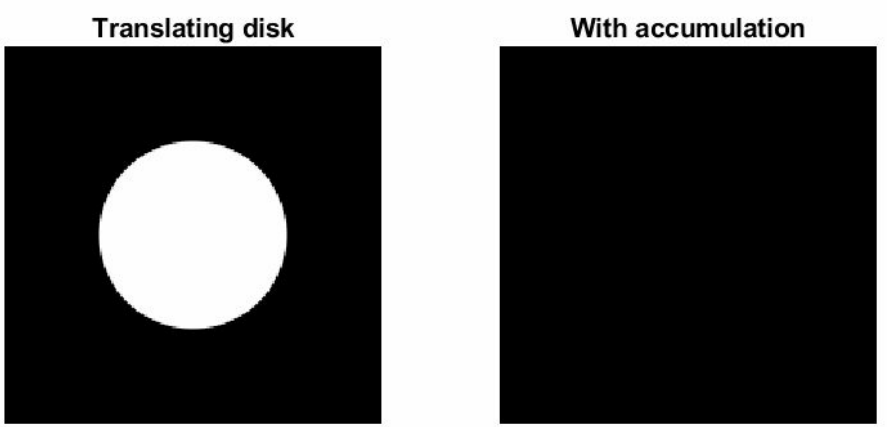
$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f}$$

$$\sigma = \rho \frac{fs}{2N}\left(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}\right)$$

$$\arg\min_{H_r} \{G_l * H_r - G_r * H_r^f\}$$

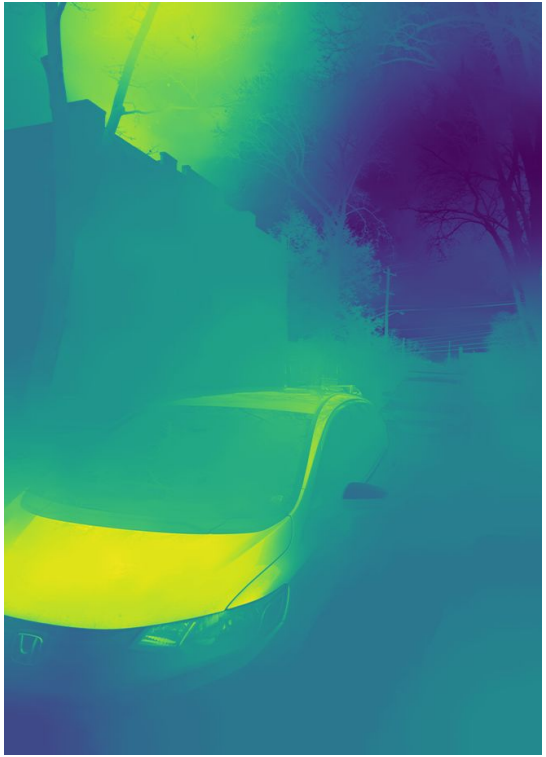
There's more!

- At a region of constant depth, can parameterize H by just the radius
  - Typically something like a Gaussian kernel
  - Proposed translating disk kernel is more realistic and accurate



$$\mathbf{H}_{r_s} = \sum_{i=0}^{|2s|} \left\{ \mathbf{C}(0,0) \odot \mathbf{C}\left(\frac{s}{|s|} i, 0 \right) \right\}$$

Results



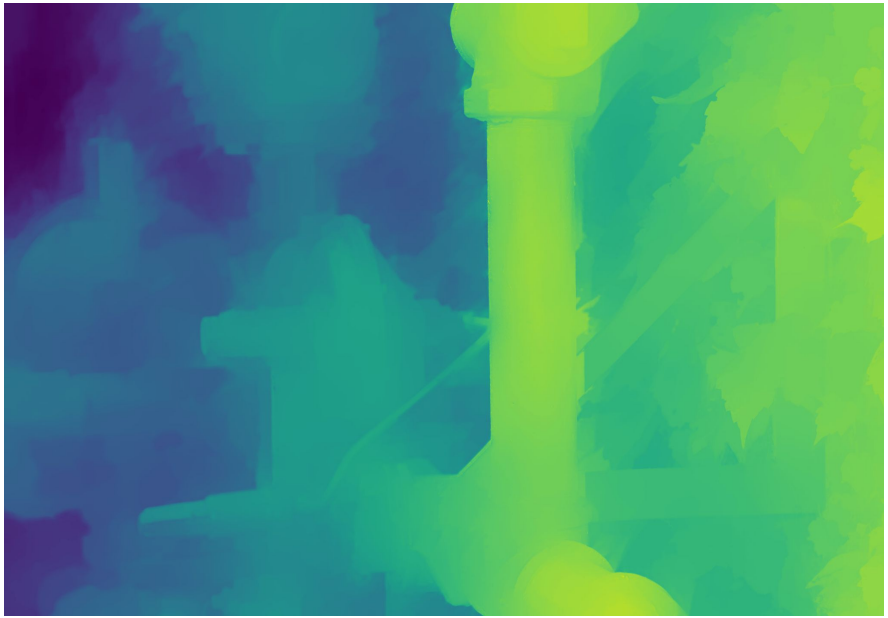


Results



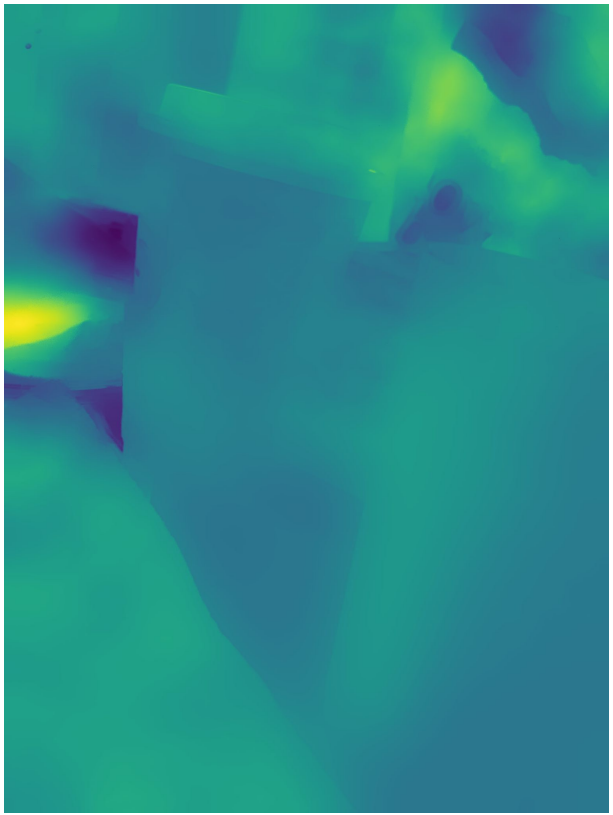


Results





Results





Discussion

- Advantages
  - Impressive that the method works on smartphone data
  - Looks reasonably good
  - Images taken rather arbitrarily, did not have to capture any training data (calibration experiment earlier was just to prove the symmetry property)
- Disadvantages
  - As with other defocus methods, struggles with untextured surfaces
  - Sliding window optimization approach is O(number of constant depth windows), which is pretty slow
  - Can be sped up using a CNN

References

[1] Abhijith Punnappurath, Abdullah Abuolaim, Mahmoud Afifi, and Michael S. Brown. Modeling defocus-disparity in dual-pixel sensors. In IEEE International Conference on Computational Photography (ICCP), 2020.

[2] Shumian Xin, Neal Wadhwa, Tianfan Xue, Jonathan T. Barron, Pratul P. Srinivasan, Jiawen Chen, Ioannis Gkioulekas, and Rahul Garg. Defocus Map Estimation and Deblurring from a Single Dual-Pixel Image. In ICCV 2021. https://imaging.cs.cmu.edu/dual_pixels/

[3] F. Mannan and M. S. Langer, “Blur calibration for depth from defocus,” in Conference on Computer and Robot Vision (CRV), 2016, pp. 281–288.

[4] T. Xian and M. Subbarao, “Depth-from-defocus: Blur equalization technique,” in SPIE: Society of Photo-Optical Instrumentation Engineers, 2006

Thank you!
Questions?

PatchMatch and Content-Aware Image Editing

By: Alan Hsu
(15-463 Fall 2021)

PatchMatch Algorithm



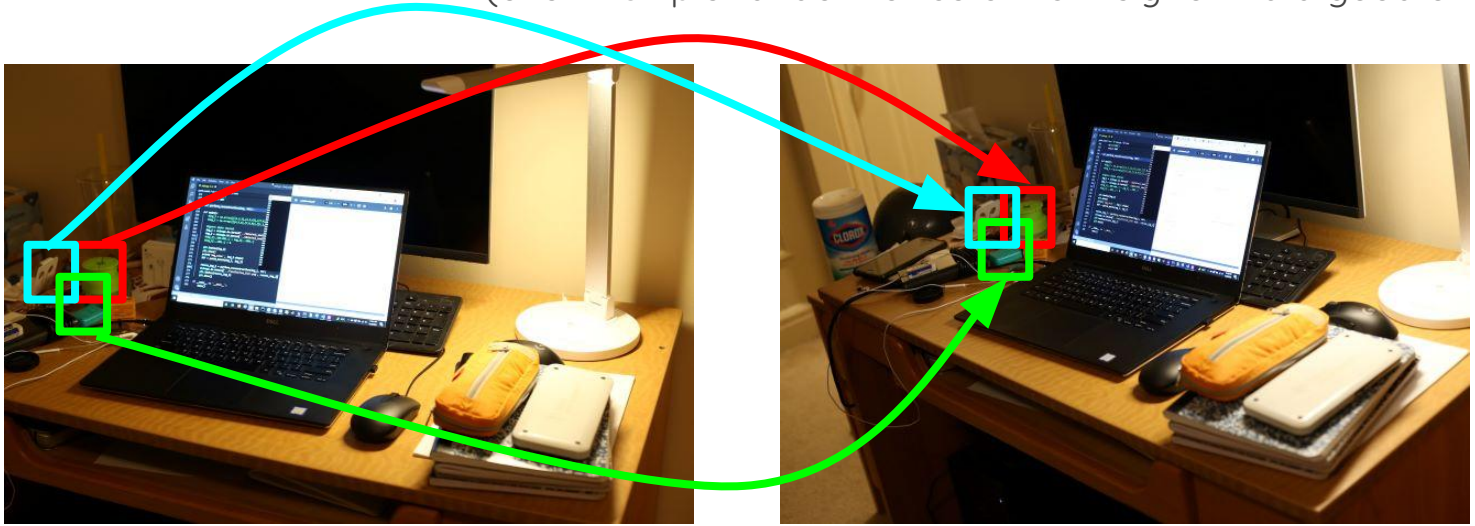
What is PatchMatch?

PatchMatch is an algorithm to produce a dense per-pixel correspondence (**Nearest Neighbor Field**, or NNF) between two images

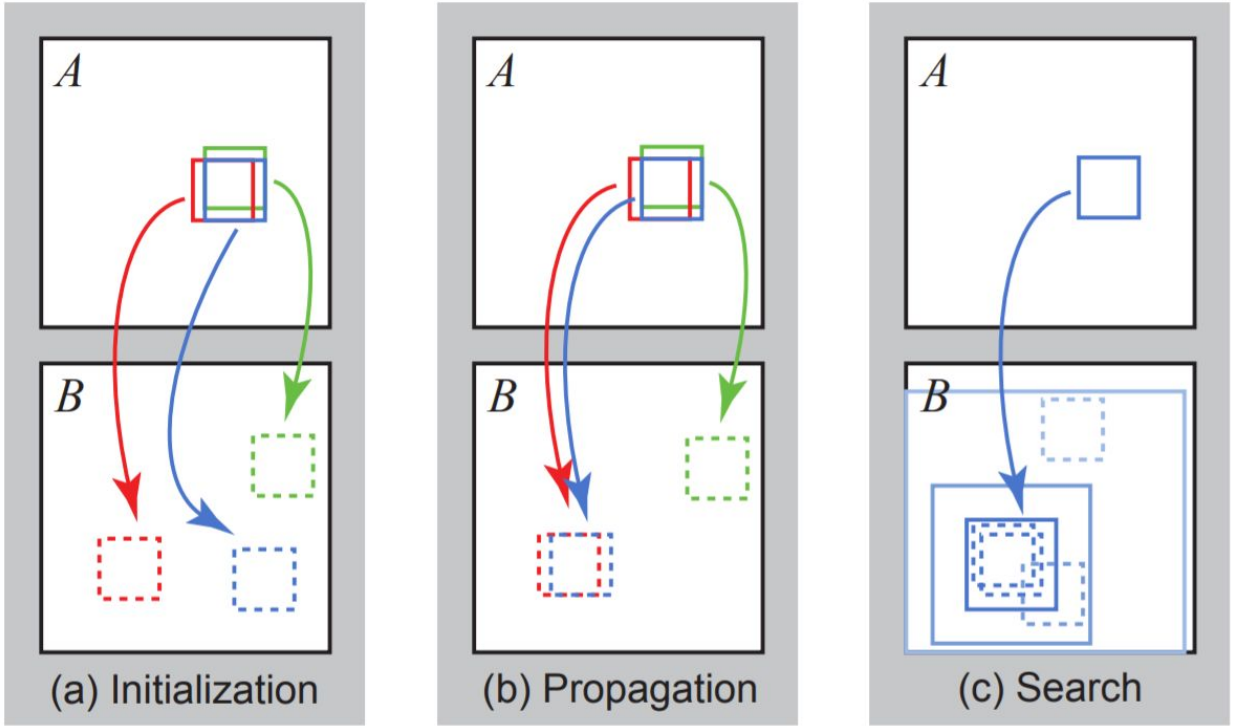


How is Patch Match more efficient?

1. Dimensionality of Offset Space (much smaller)
2. Natural Structure of Images
(adjacent pixels likely have a similar offset)
3. Law of Large Numbers
(over multiple random offsets it is likely to find a good offset)

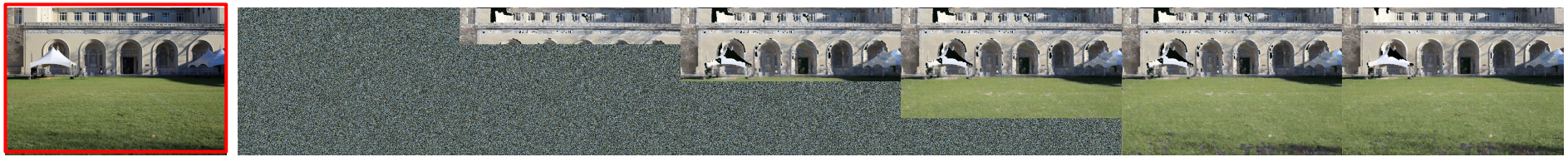


How does PatchMatch work?



Example of Convergence

The top image is reconstructed using the pixels from the bottom image





Iter 0 Iter 0.25 Iter 0.5 Iter 0.75 Iter 1 Iter 5

PatchMatch Applications





Can you spot the difference?





How about this one?

We compute the NNF from the hole to the rest of the image!









Guided Image Completion

More Structural Image Editing Results








Per-Pixel Style Transfer



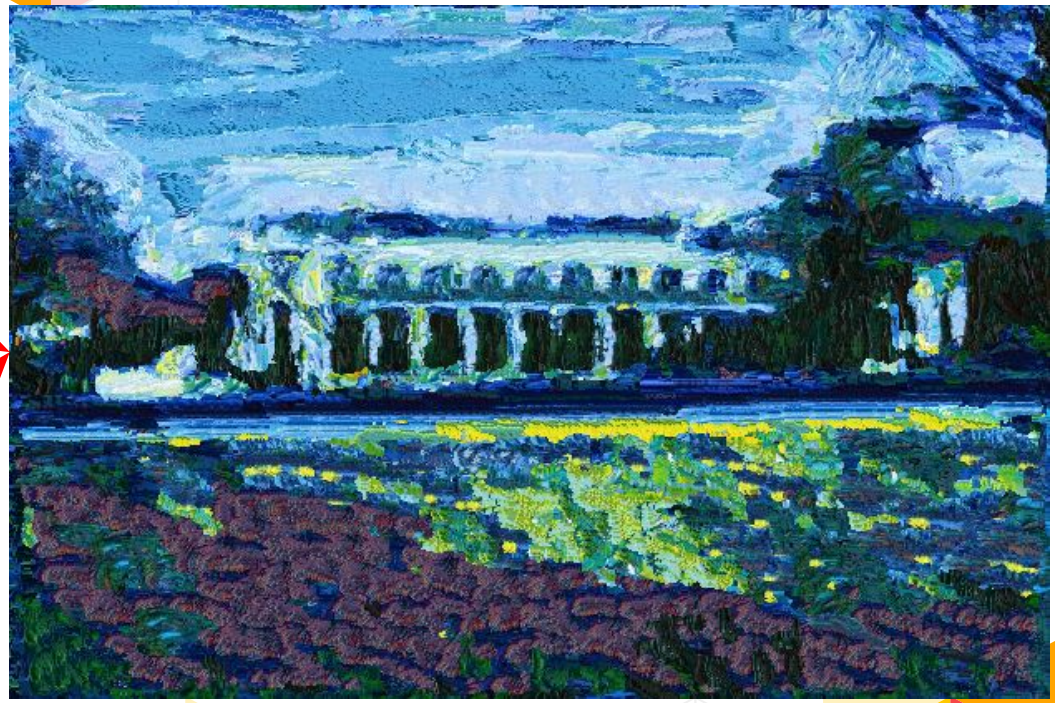




Oil Painting!



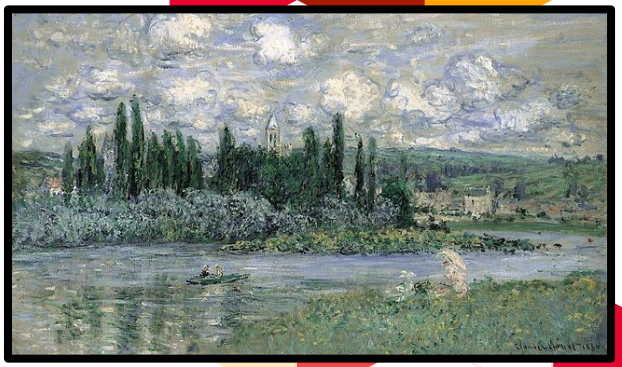




More Painting Results (Monet)











Citations and Acknowledgements:

1.) **Prof. Ioannis Gkioulekas** for the all the help he provided!

2.) **PatchMatch paper:** https://gfx.cs.princeton.edu/pubs/Barnes_2009_PAR/patchmatch.pdf

3.) **SIFT Flow:** https://people.csail.mit.edu/celiu/SIFTflow/

4.) **Style Transfer:** https://arxiv.org/pdf/1508.06576.pdf

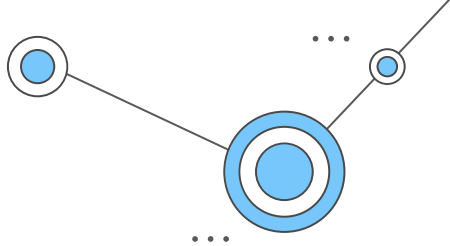
5.) **Vincent van Gogh** and **Claude Monet** for their paintings

Thanks!

DiffuserCam for 3D Printing Applications

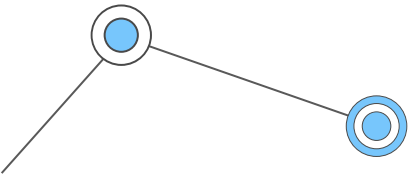
Joon Jang, jiwoong@andrew.cmu.edu

DiffuserCam

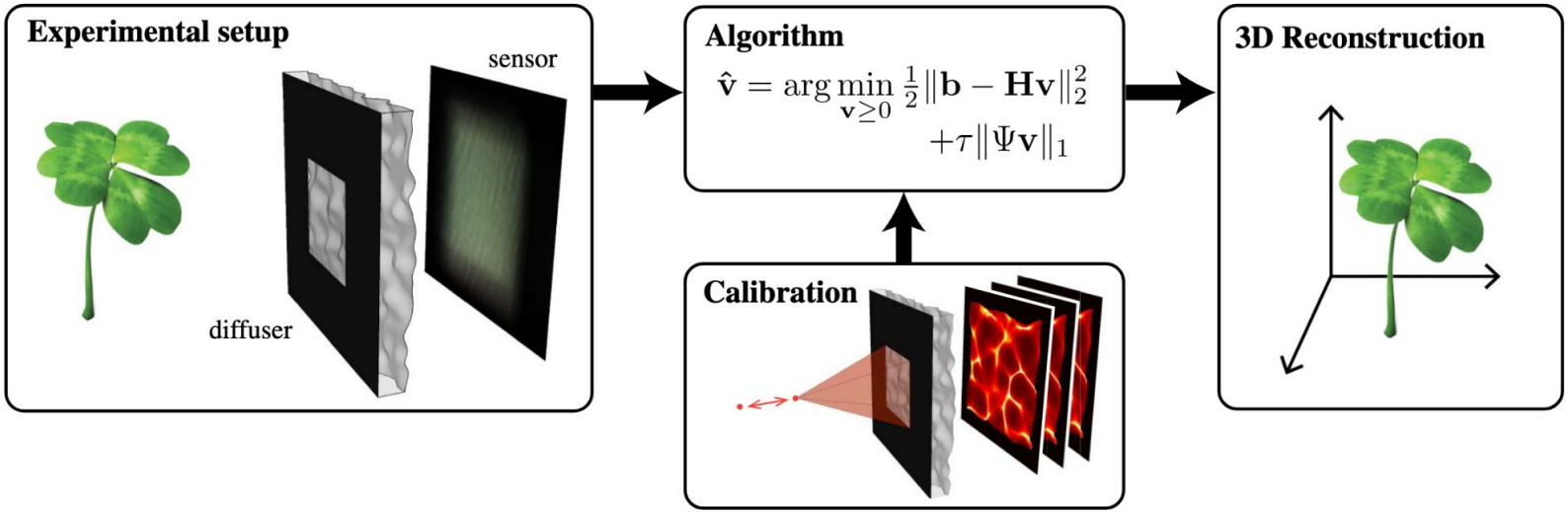


DiffuserCam is a compact and relatively simple computational camera for single-shot 3D imaging.

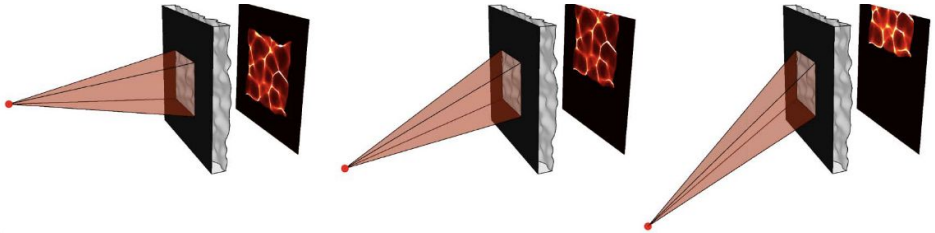
Due to the single-shot nature and affordability of components to build it, I'm exploring whether it'd be usable in 3D printing contexts.



How it Works

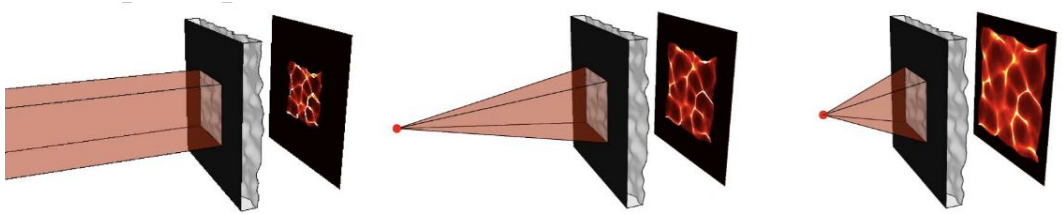

Experimental setup
sensor
diffuser
Algorithm
$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}\geq 0} \frac{1}{2}\|\mathbf{b} - \mathbf{H}\mathbf{v}\|_2^2 + \tau\|\Psi\mathbf{v}\|_1$
Calibration
3D Reconstruction

Theory: PSF Differences

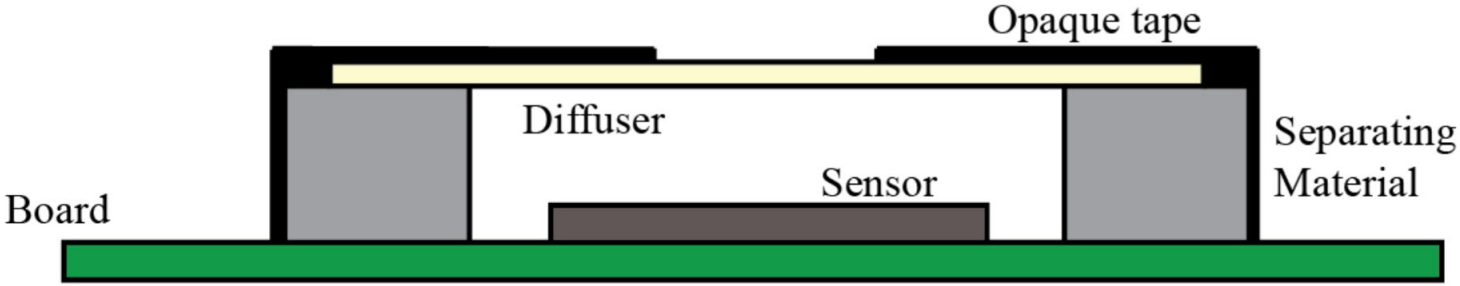


Lateral changes leads to changes in placement in the PSF on the sensor.

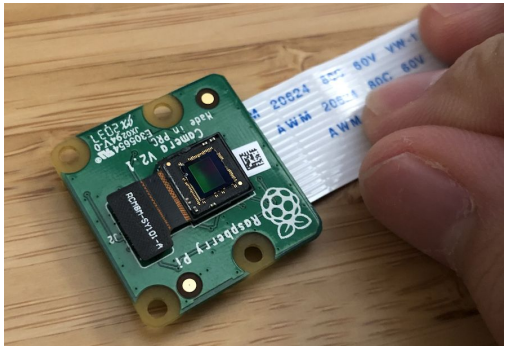
Similarly, Vertical changes leads to changes in placement in the PSF on the sensor.

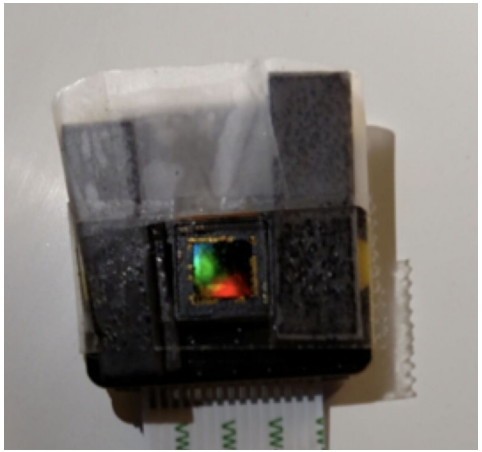


How It's Made

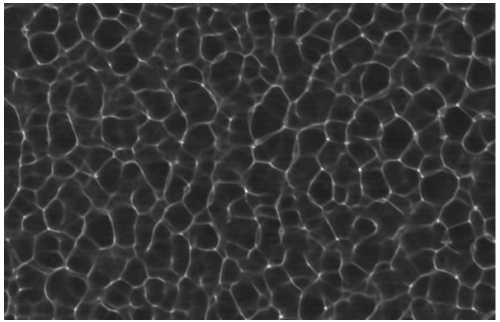


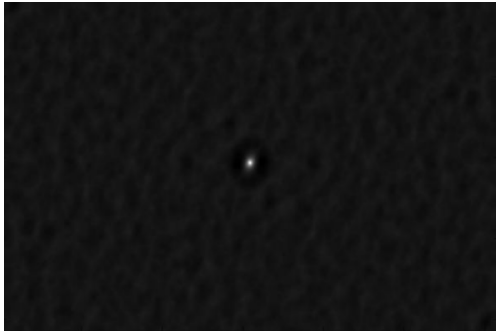
How It’s Made

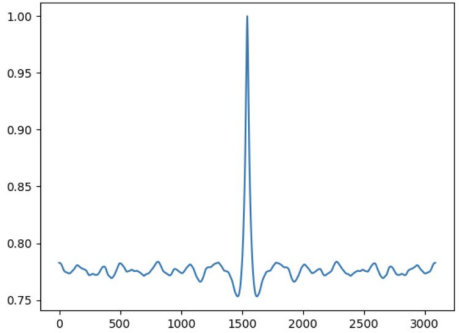




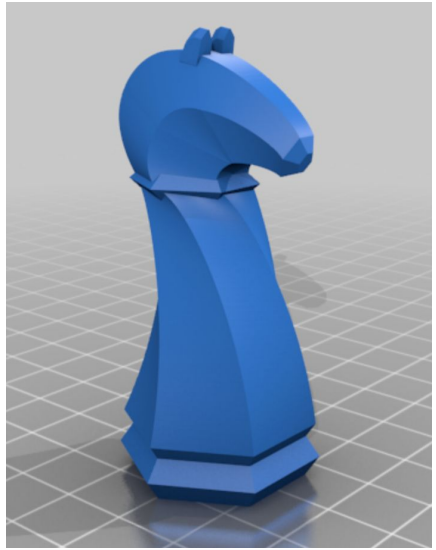
Calibration







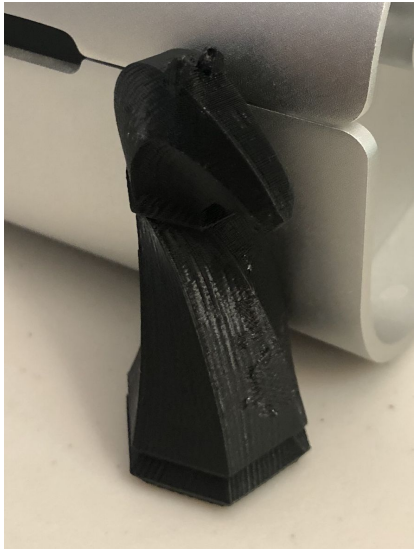
Preliminary Results



3D File

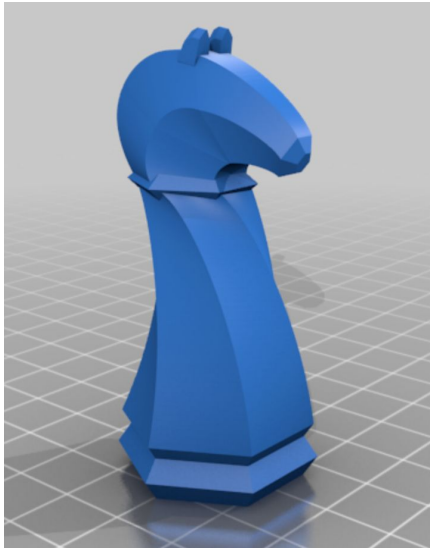


3D Print



Reconstructed Print

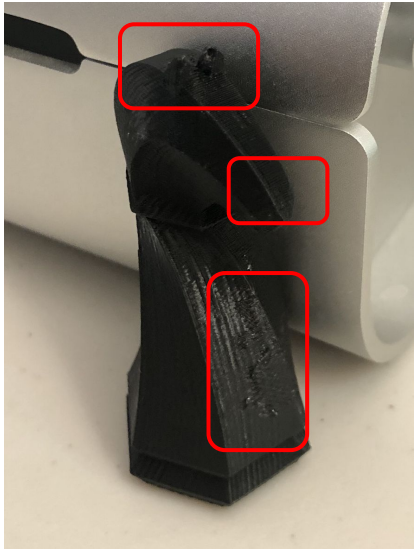
Preliminary Results



3D File



3D Print

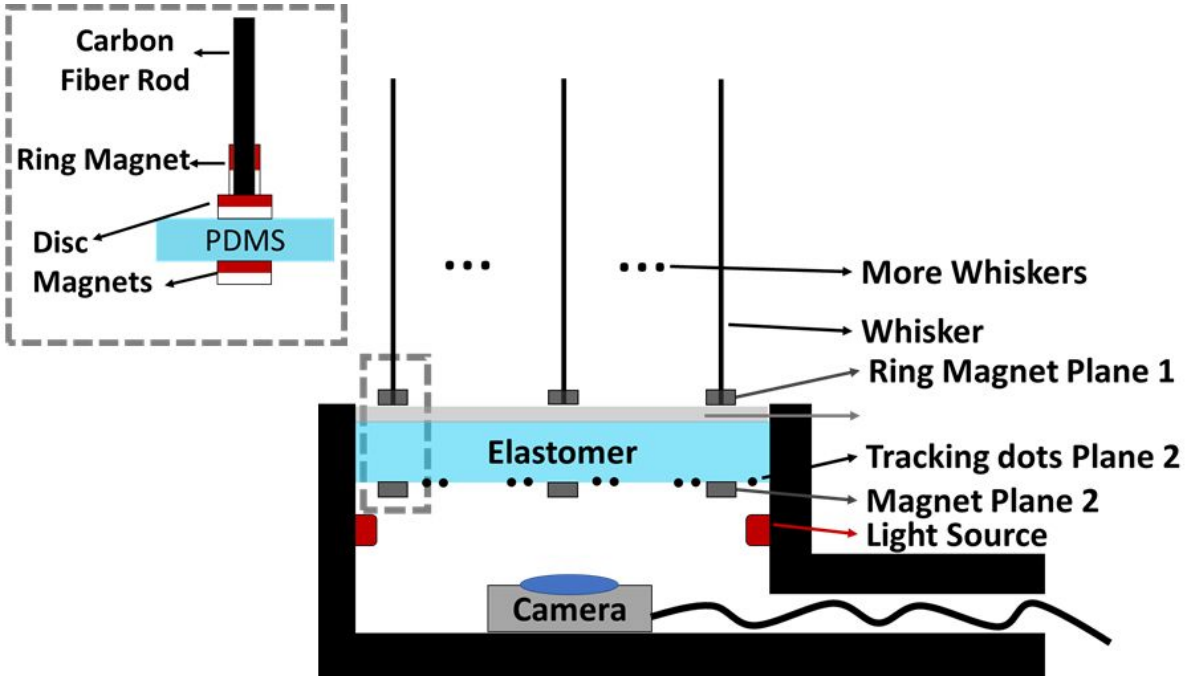


Reconstructed Print

Applying Computational Photography techniques to the WhiskSight Sensor

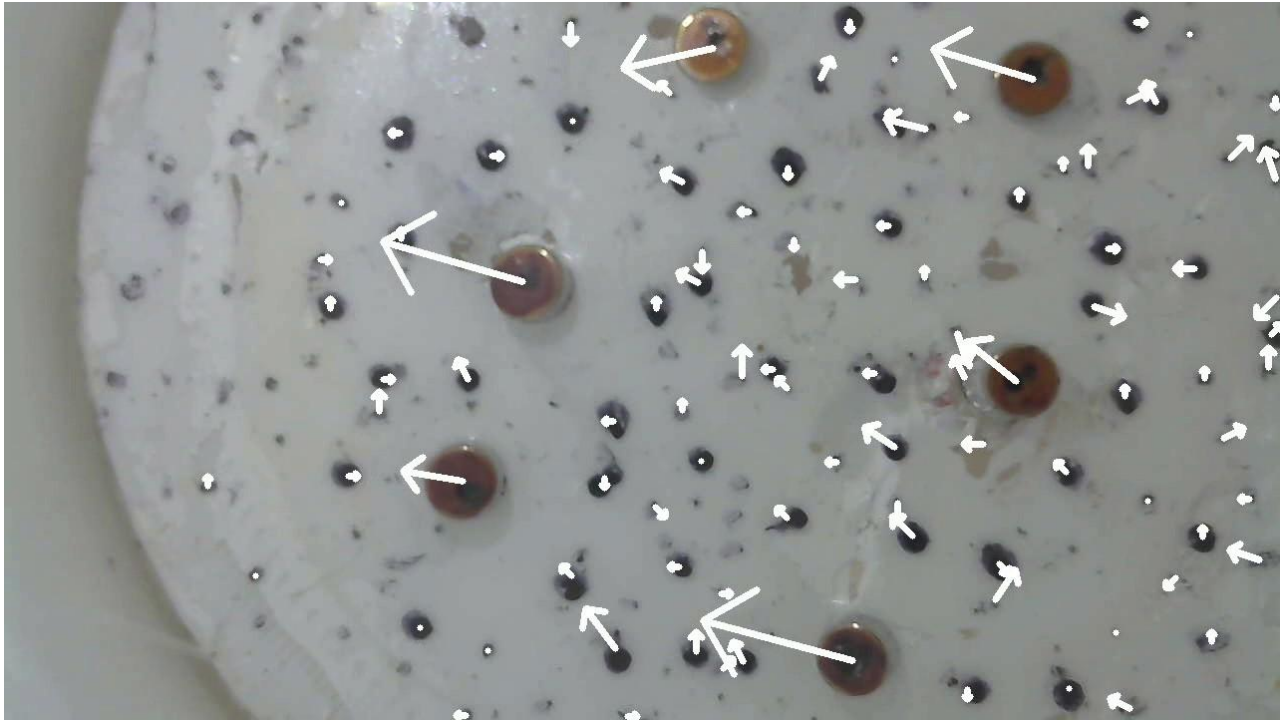
Teresa Kent

WhiskSight Sensor



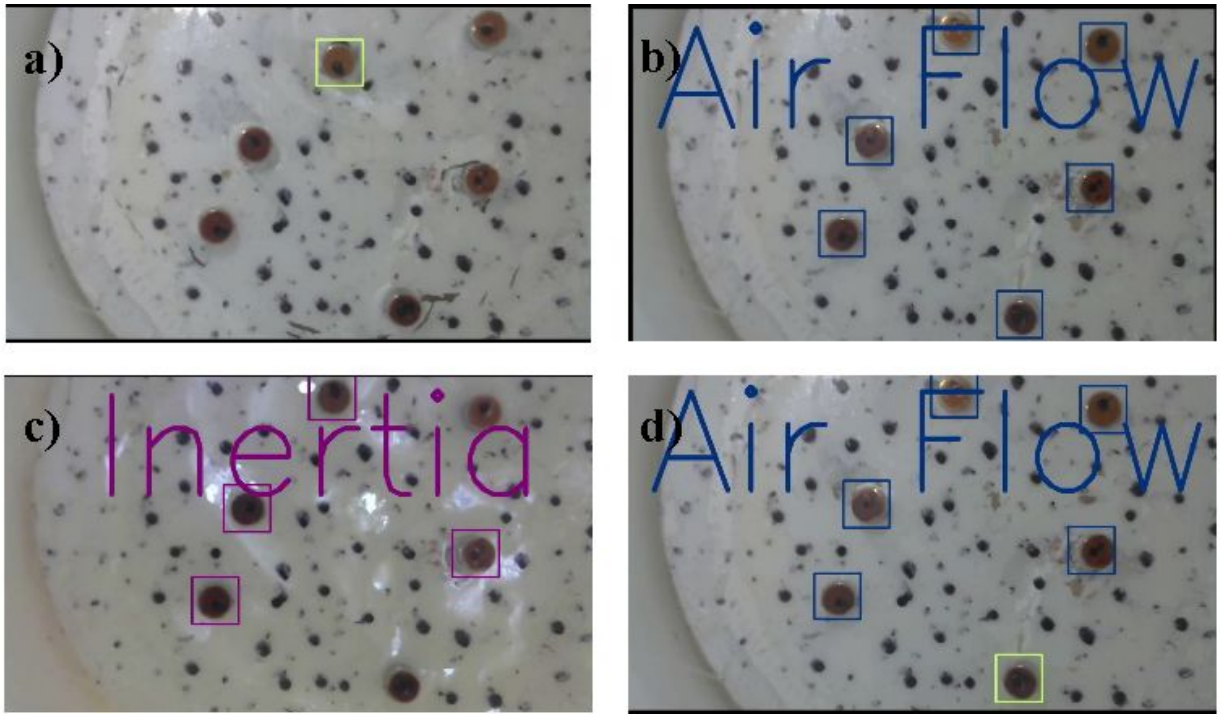
Kent, Teresa A., et al. "WhiskSight: A Reconfigurable, Vision-Based, Optical Whisker Sensing Array for Simultaneous Contact, Airflow, and Inertia Stimulus Detection." *IEEE Robotics and Automation Letters* 6.2 (2021): 3357-3364.

What the Camera Sees



Kent, Teresa A., et al. "WhiskSight: A Reconfigurable, Vision-Based, Optical Whisker Sensing Array for Simultaneous Contact, Airflow, and Inertia Stimulus Detection." *IEEE Robotics and Automation Letters* 6.2 (2021): 3357-3364.

Current Application



Kent, Teresa A., et al. "WhiskSight: A Reconfigurable, Vision-Based, Optical Whisker Sensing Array for Simultaneous Contact, Airflow, and Inertia Stimulus Detection." *IEEE Robotics and Automation Letters* 6.2 (2021): 3357-3364.

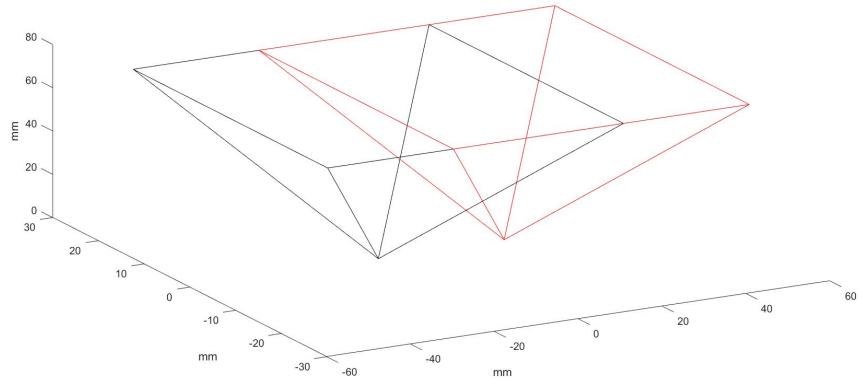
What Improvements can be made

Handling situations which break the tracker

Tradeoffs between array size and accuracy

Novel Applications

Understanding the Camera Space/Overlap



Two Camera's View of the 3d world

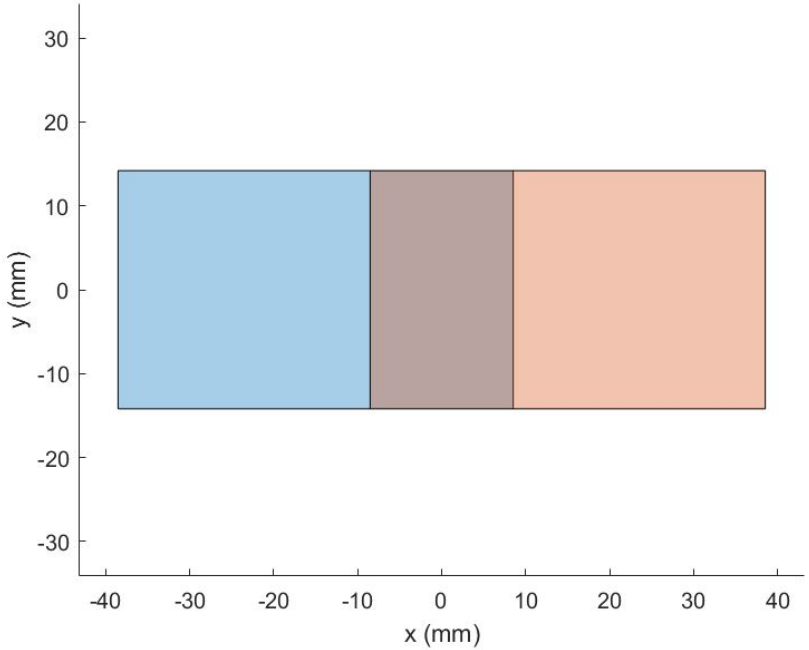
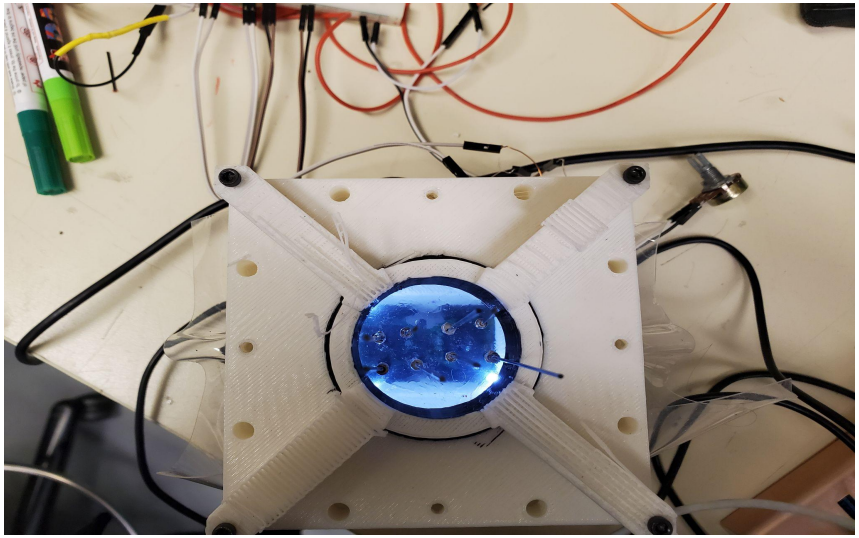
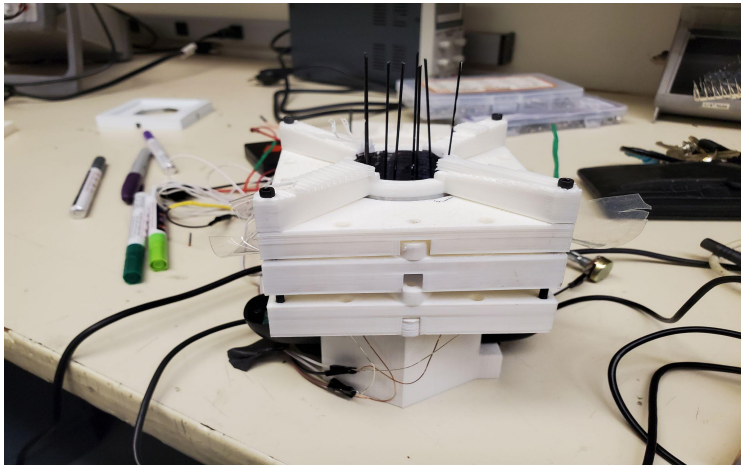
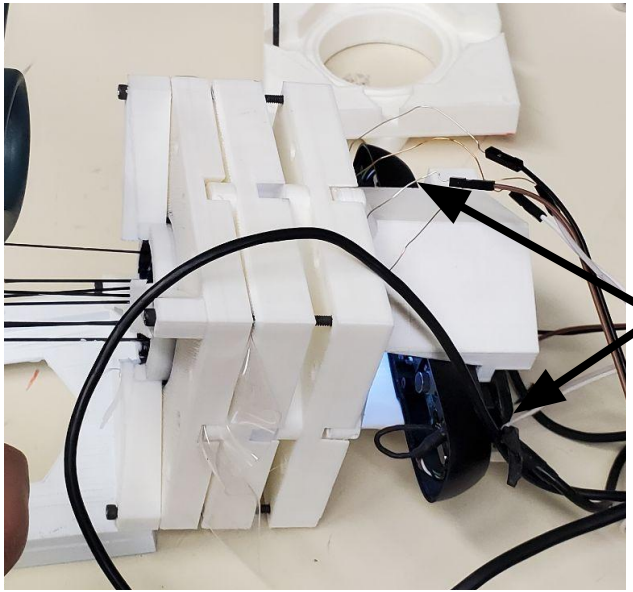


Image Overlap at 50 mm

Updated Sensor







Visual Representation of the Overlapping Space

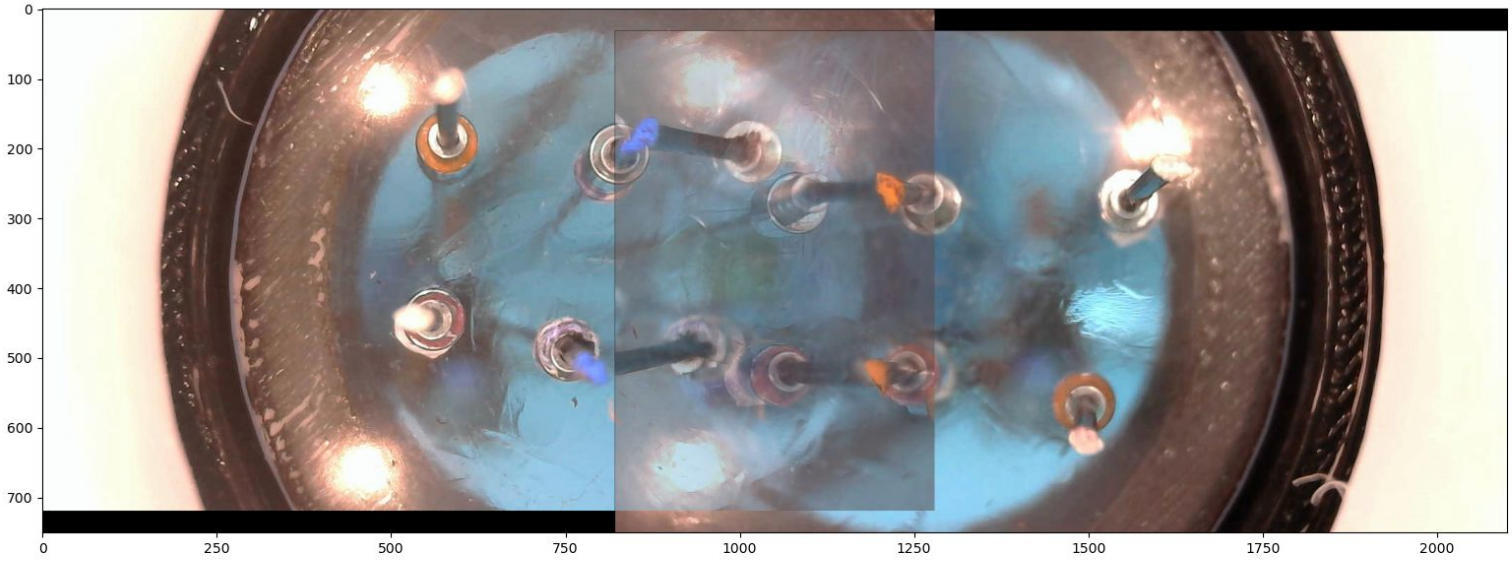
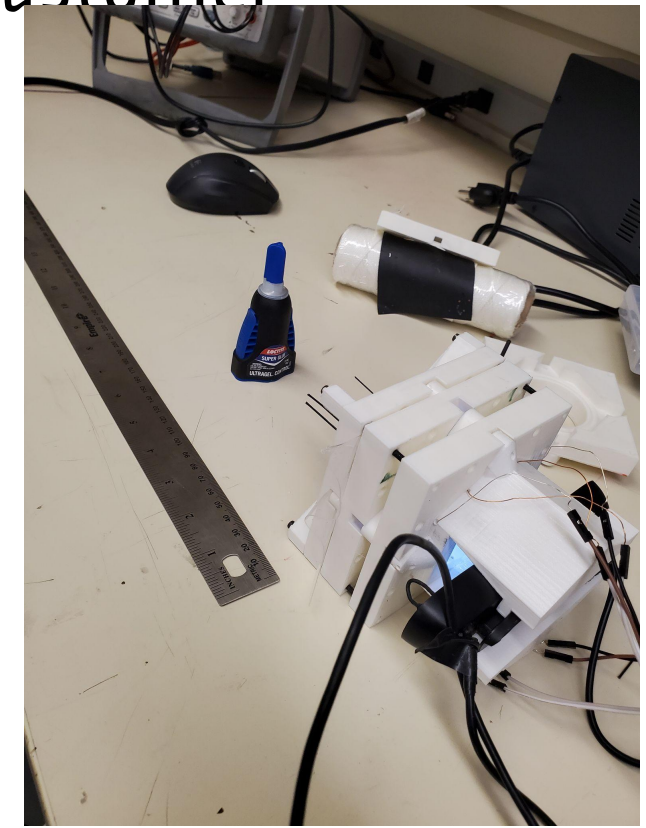
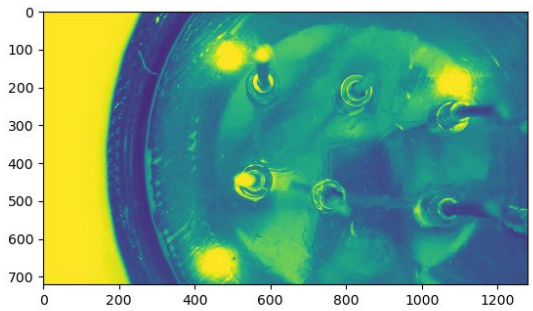


Image Detection Through the Elastomer



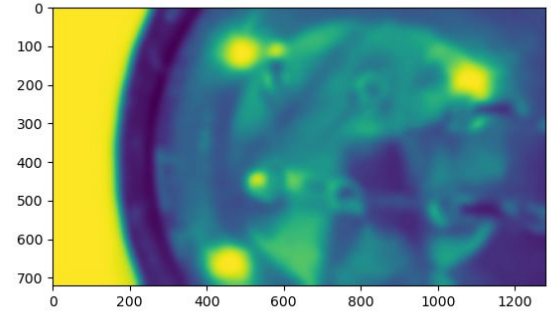


Use the sharpness to distinguish the objects in front of and behind the elastomer



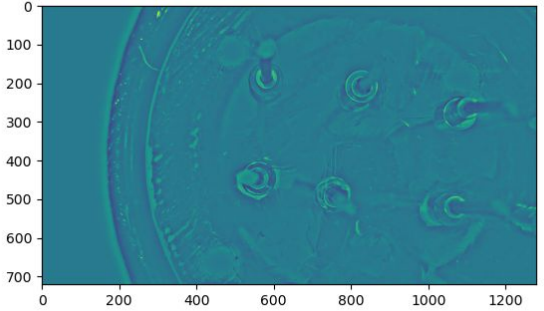
Luminance

-



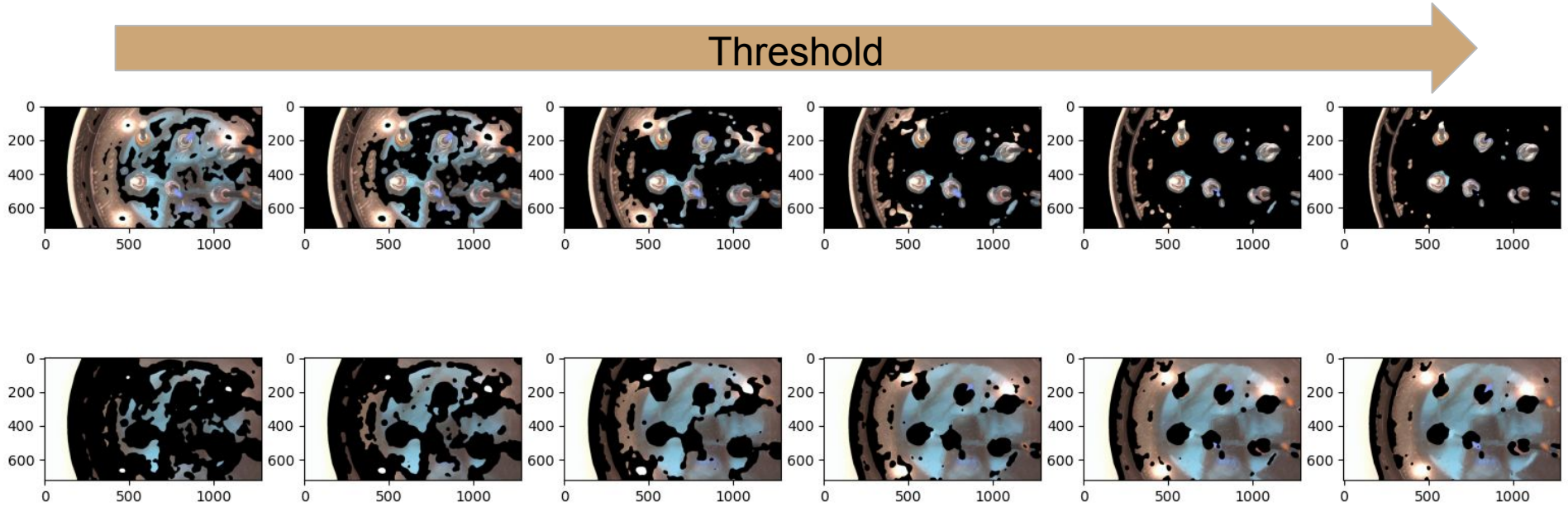
Blurred Image

=

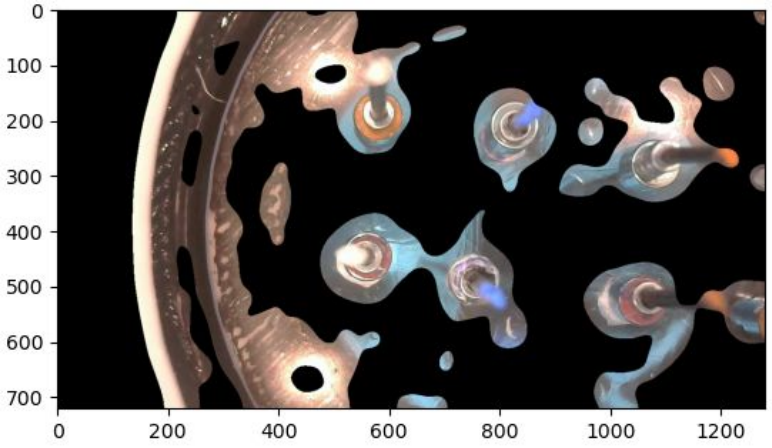


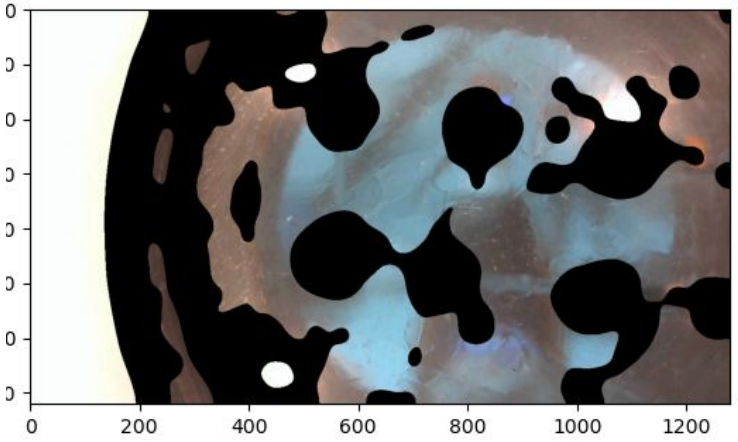
High Detail Image

Tested Different Thresholds for Sharpness and Sigma Values

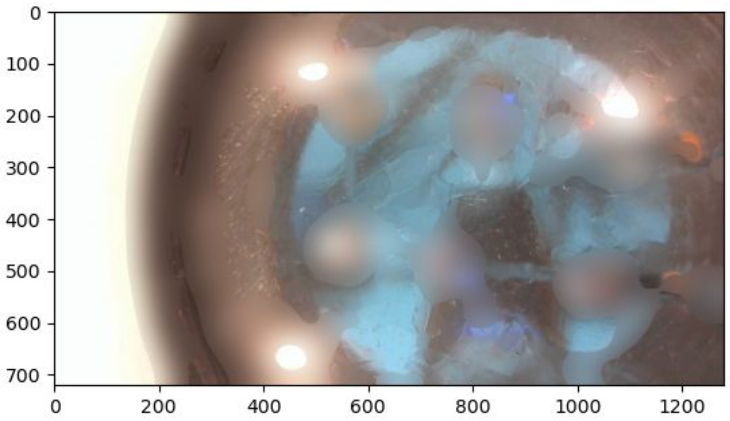


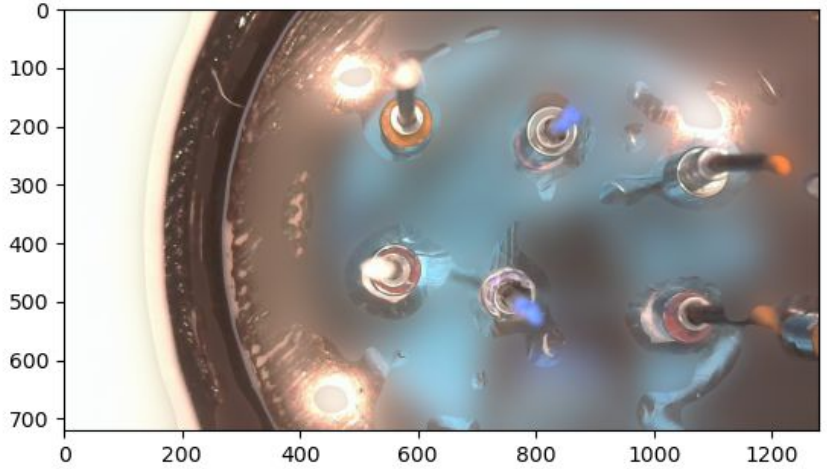
Separated Image





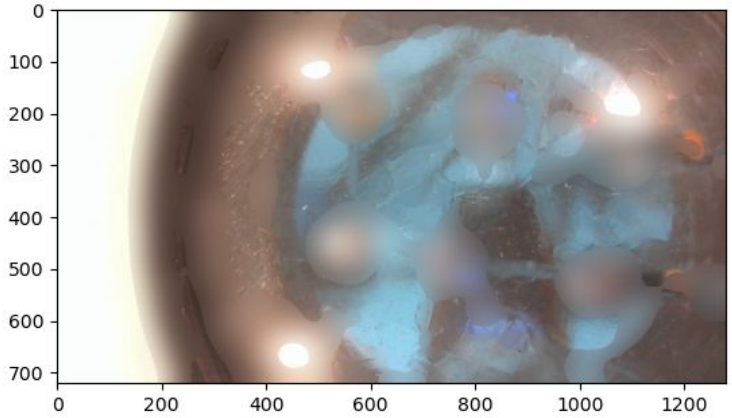
Separated Image





What's Next

- Create 3d Map of the external scene
- Deblur the external scene

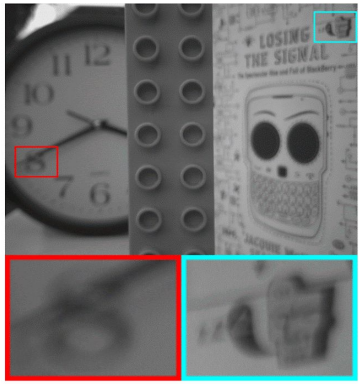


- Quantize the improvement in localization of the whisker using two cameras.

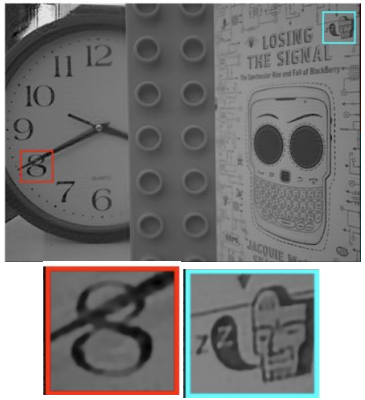
Huixuan Tang, Scott Cohen, Brian Price, Stephen Schiller and Kiriakos N. Kutulakos, Depth from defocus in the wild. *Proc. IEEE Computer Vision and Pattern Recognition Conference*, 2017.

Heide, F., Rouf, M., Hullin, M. B., Labitzke, B., Heidrich, W., & Kolb, A. (2013). High-quality computational imaging through simple lenses. *ACM Transactions on Graphics (TOG)*, *32*(5), 1-14.

Thank You



DP image



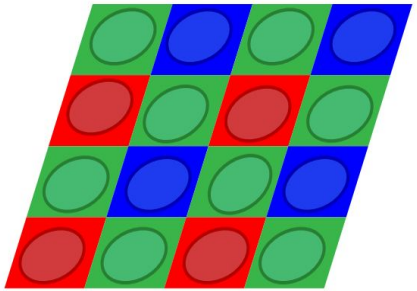
Expected all-in-focus image

All-in-focus Image Estimation from Dual Pixel Images

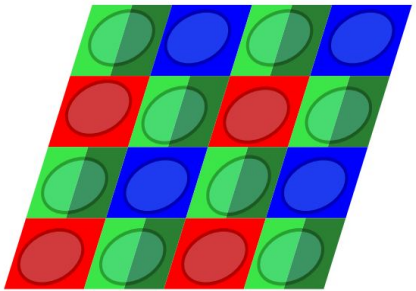
Akash Sharma Tarasha Khurana

What are DP images and why we care about all-in-focus?

Each pixel of the **Dual-Pixel sensor** is divided into two parts, **left and right**. It captures images much like a **stereo camera** with a small baseline.



a) Traditional Sensor



b) Dual-Pixel Sensor

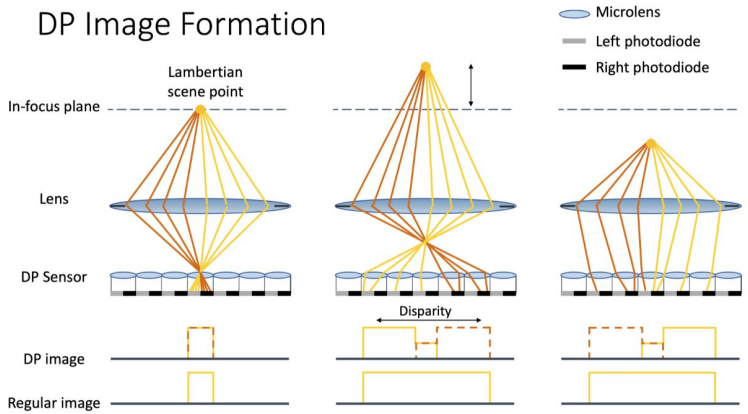


Some devices with Dual Pixel sensors

All-in-focus images are used extensively in robotics applications. Existing robotics algorithms cannot run on defocus-blurred images.

Can we recover an all-in-focus image from a DP image?

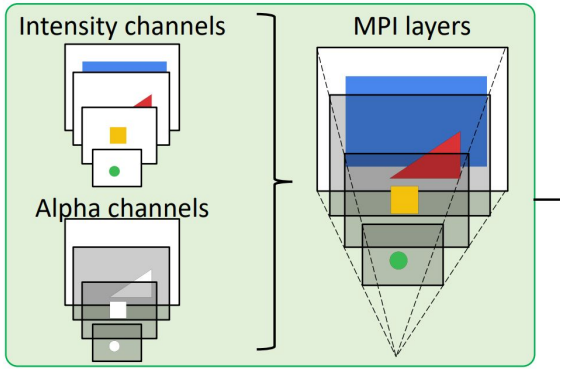
Dual Pixel Images are **two-sample light fields** and each pixel integrates light from **half the main lens** aperture.



Idea: Use disparity of out-of-focus points as a guide to defocus the image. Problem is still **under-constrained**.

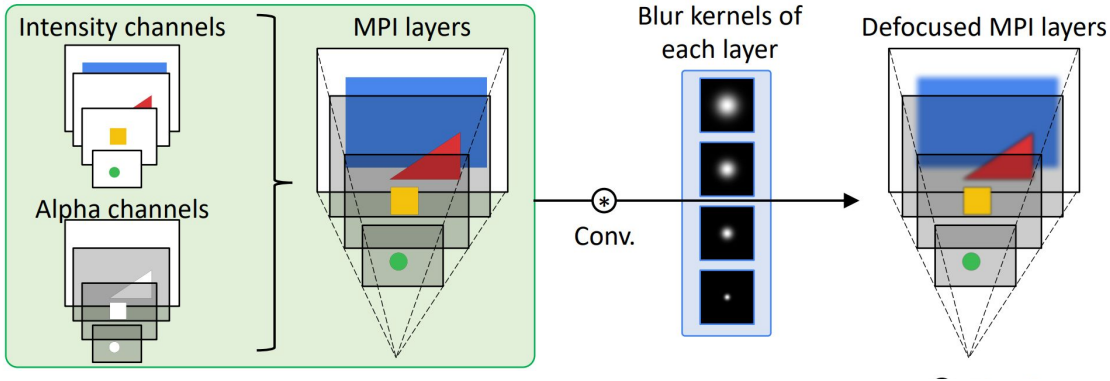
How has this been done before?

Model the dual pixel (left, right) images as an observation rendered from an underlying **Multiplane Image.**



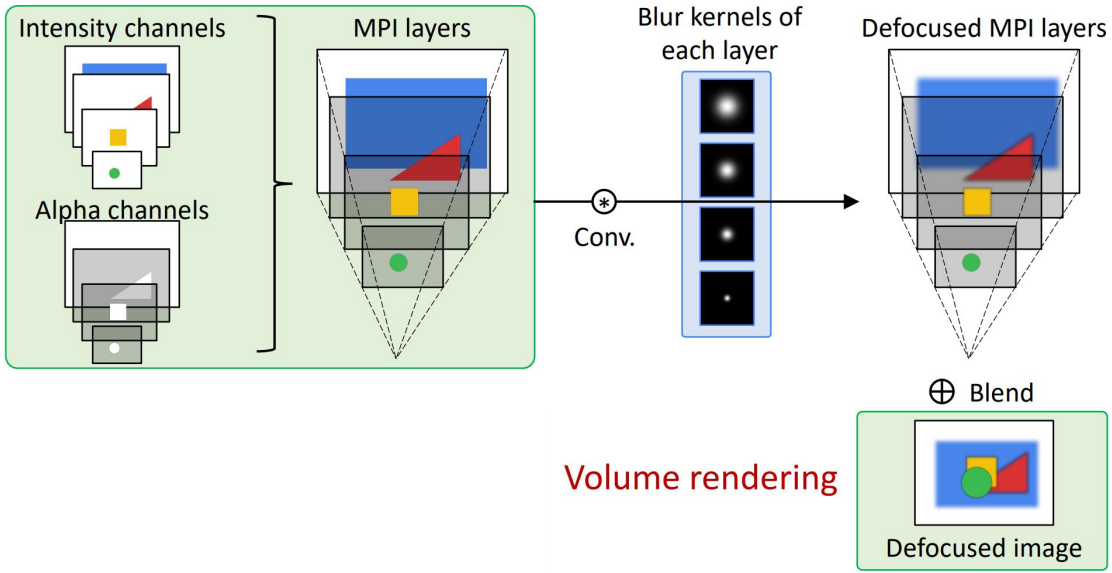
How has this been done before?

Model the dual pixel (left, right) images as an observation rendered from an underlying **Multiplane Image.**



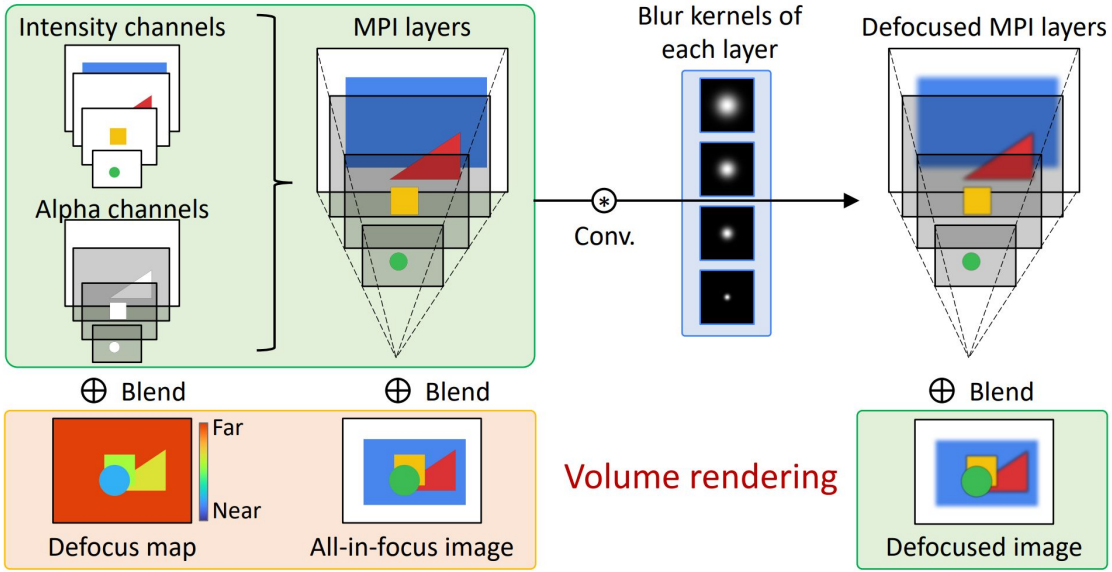
How has this been done before?

Model the dual pixel (left, right) images as an observation rendered from an underlying **Multiplane Image.**



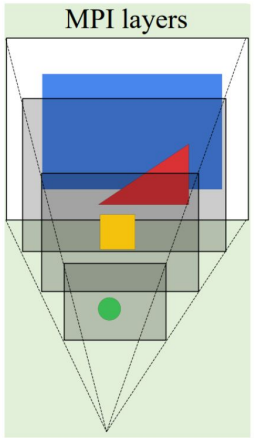
How has this been done before?

Model the dual pixel (left, right) images as an observation rendered from an underlying **Multiplane Image.**

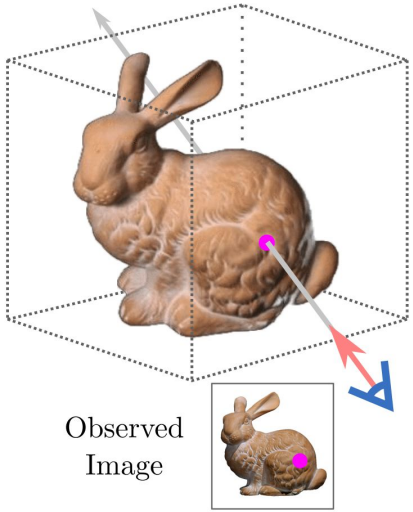


What are some limitations that can be addressed?

- MPIs are good at **rendering images fast** but cannot represent scenes with **continuous depths**.
- We can rather render defocus image observations from continuous representations such as **Neural Radiance Fields**.



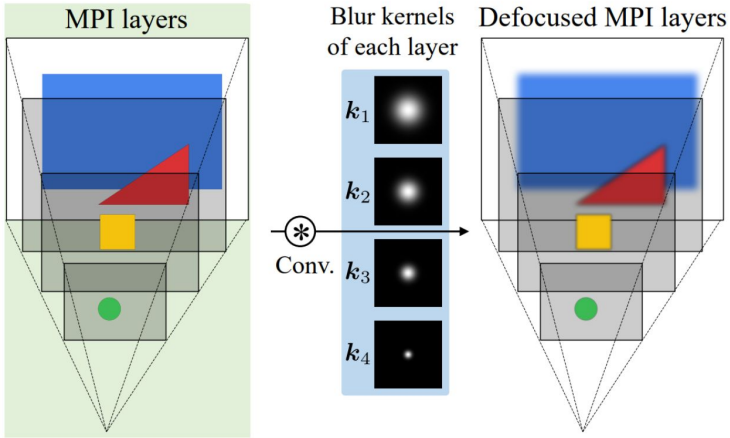
Discrete depth representations can model only coarse relative ordering of pixels



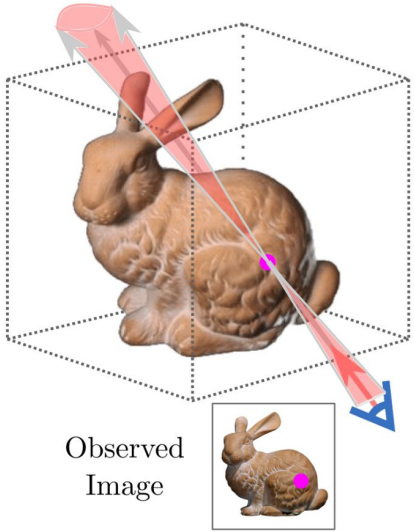
Model defocus image using rendering from continuous volumetric representation.

End of Story? Why is this difficult?

- With MPI you can blur image planes using **2D convolution** with lens kernels.
- With continuous representation, it is **unclear** how lens aberrations and spatially varying blur of lenses can be similarly convolved or modeled



Easy to model blurring



Trace hourglass like cones through volume and weight regions continuously.

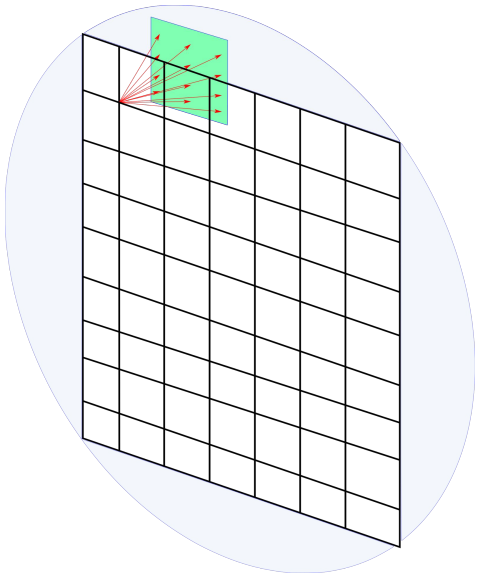
What is a workaround? *Thin-lens rendering*

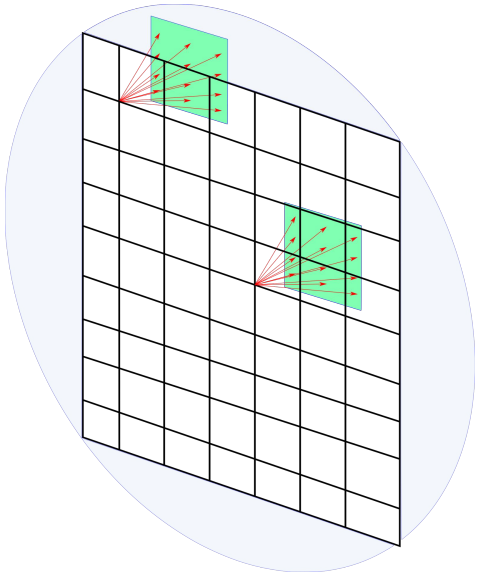
- Bypass blurring altogether by sampling the lens aperture as an *n x m* grid i.e., **render a lightfield.**
- For a NeRF[1], this means rendering an image at each of these *n x m* locations on the lens.
- Then **average left and right half** of the lens aperture to develop a rendering model for dual pixel images

Lens aperture

Image plane

Rays from aperture plane

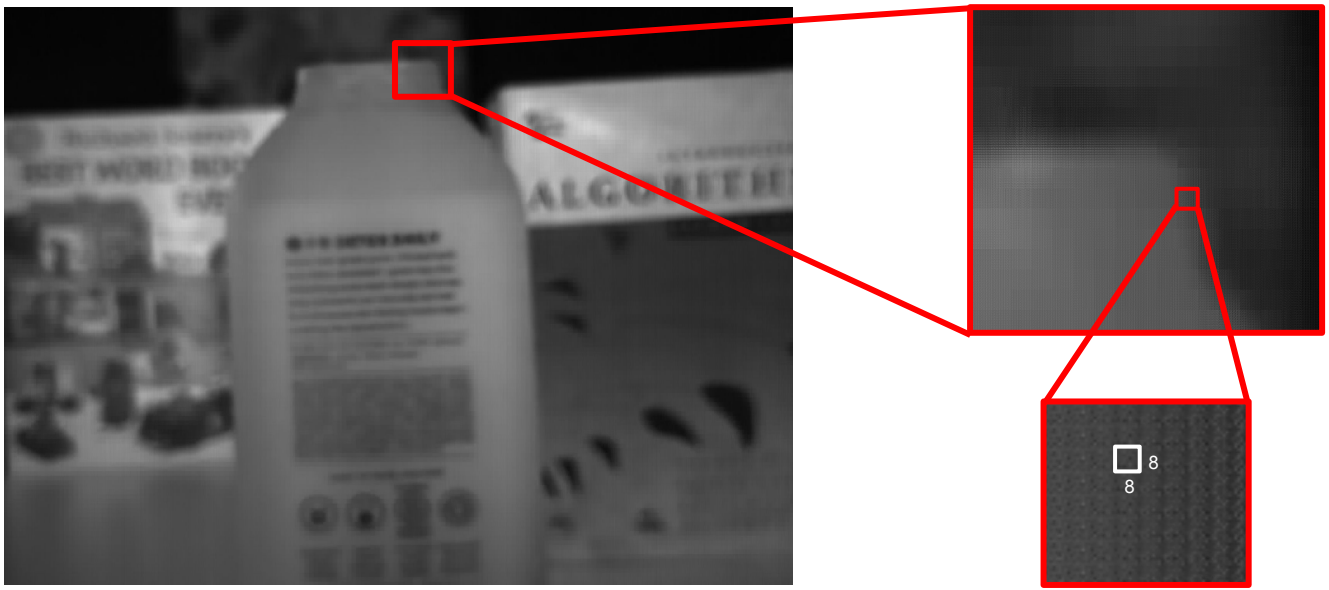




[1] We are skipping a lot of volumetric rendering details for NeRF.

So, does this work? *Kind of!*

Implementing this gives us a lightfield image where we have a **8 x 8 subaperture view** of every pixel on the image.



We can render the left and right DP images from these by averaging the 8 x 4 halves of all the pixels.

So, does this work? *Kind of!*

Although rendered dual pixel images look good (PSNR: 43.13) …

Predicted Images



Left DP Image



Right DP Image

So, does this work? *Kind of!*

Although rendered dual pixel images look good (PSNR: 43.13) …

Target Images



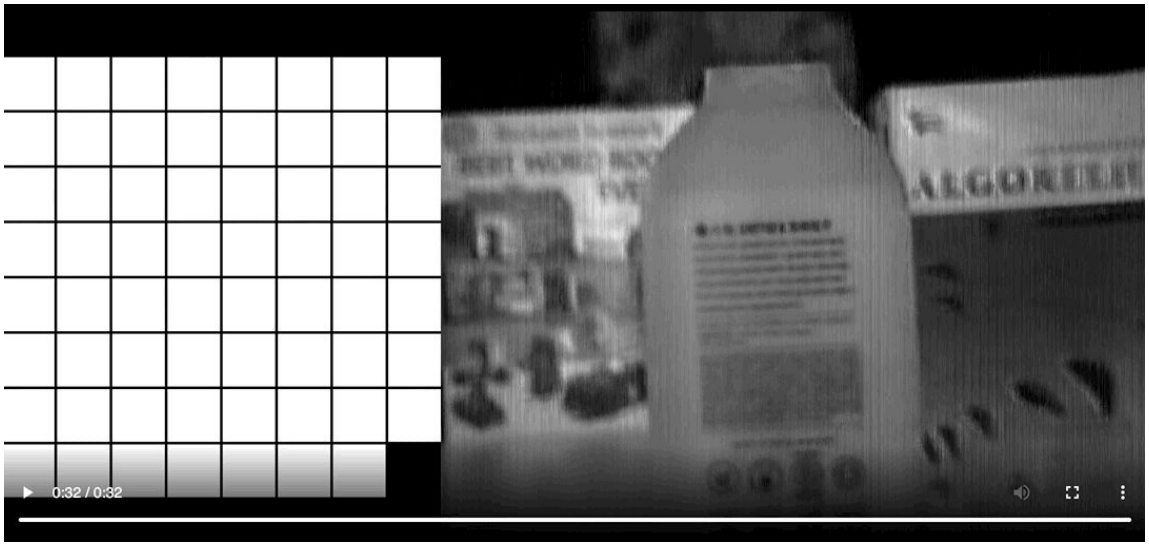
Left DP Image



Right DP Image

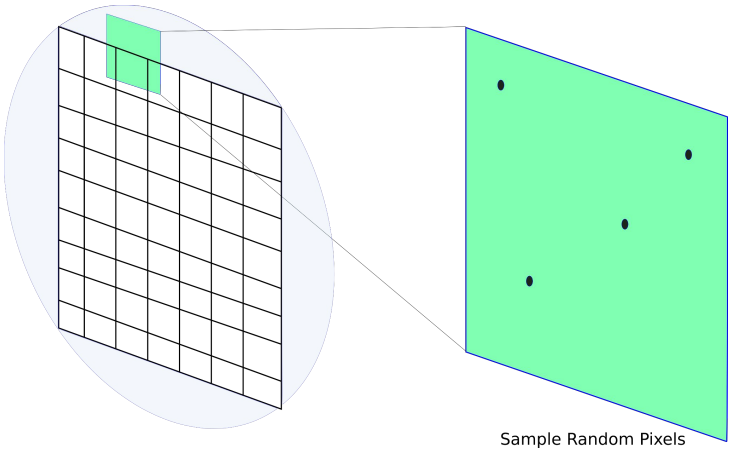
So, does this work? *Kind of!*

The individual pinhole images (or all-in-focus images) look jagged and we need a way to enforce pixel-wise smoothness.



How can we enforce smoothness?

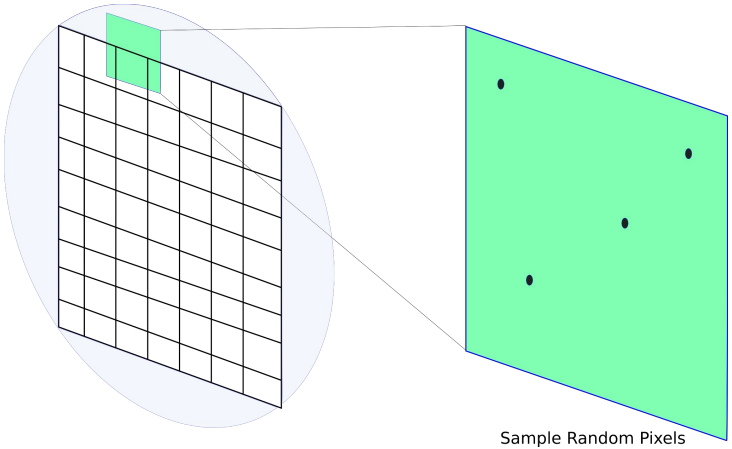
Through a regularization loss between neighbouring pixels, like the Smooth L1 loss.



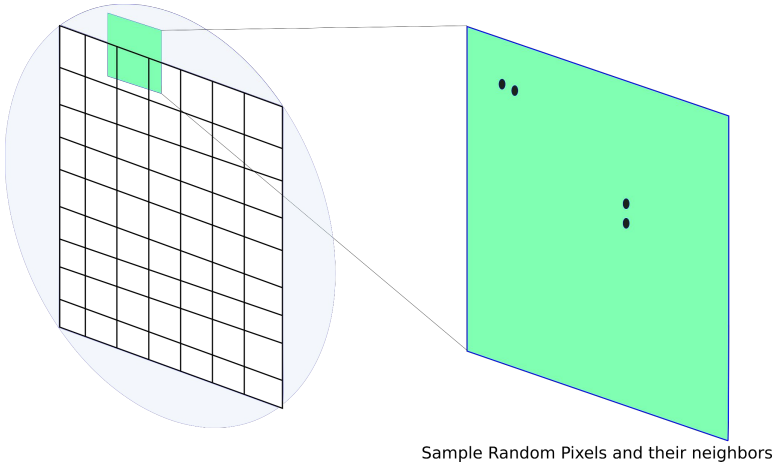
Recall that default pixel sampling in NeRF is **random**.

How can we enforce smoothness?

Through a regularization loss between neighbouring pixels, like the Smooth L1 loss.



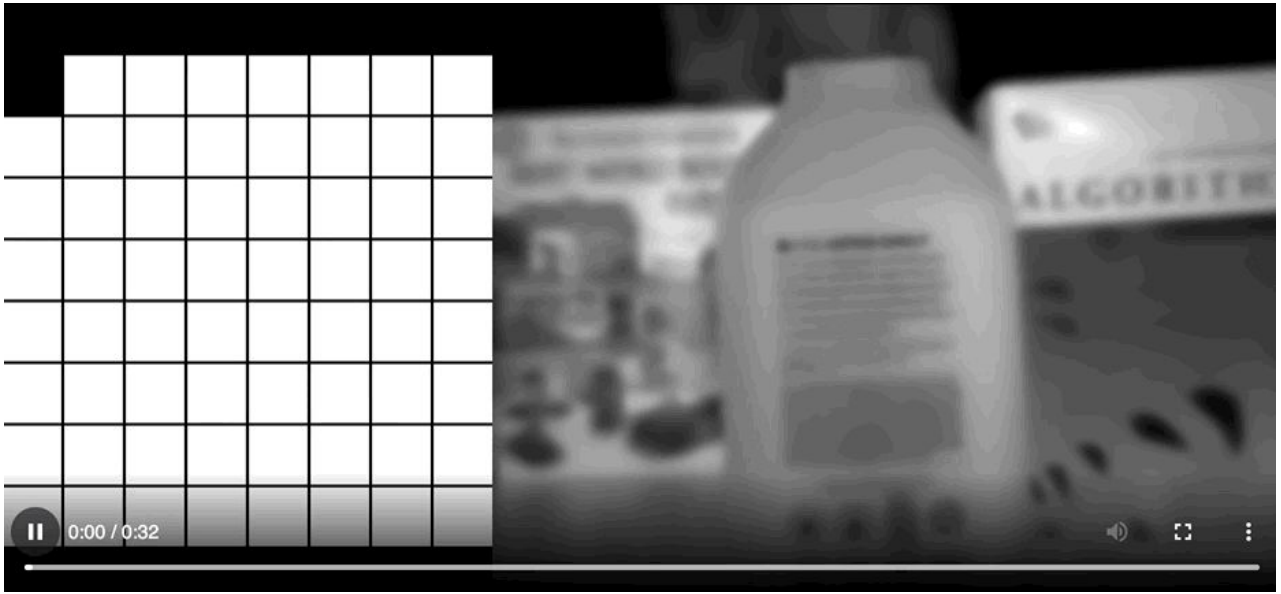
Recall that default pixel sampling in NeRF is **random**.



We modify to randomly sample only half the pixels and then add their **x- and y- neighbours** (chosen with equal probability).

Does enforcing smoothness help? *Not really*

All-in-focus images become excessively smooth (almost stretched) …



*Note that we did not have enough GPU cycles for hyperparameter tuning for any of the experiments

Does enforcing smoothness help? *Not really*

and PSNR drops on the dual pixel images from 43.13 to 32.6.

Predicted Images



Left DP Image



Right DP Image

Does enforcing smoothness help? *Not really*

and PSNR drops on the dual pixel images from 43.13 to 32.6.

Target Images



Left DP Image



Right DP Image

What is really going wrong?

- Even with a forward model defined, the problem is **under-constrained**.
- Careful **regularization** with losses is required to obtain good results.

What can still be done?

- Brute force approach, really really slow
- Improve aperture sampling by reusing ray samples
- Use an implicit network to model light field directly

References

[1] Shumian Xin, Neal Wadhwa, Tianfan Xue, Jonathan T. Barron, Pratul P. Srinivasan, Jiawen Chen, Ioannis Gkioulekas, and Rahul Garg. Defocus Map Estimation and Deblurring from a Single Dual-Pixel Image, ICCV 2021.

[2] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R. and Ng, R., 2020, August. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision* (pp. 405-421). Springer, Cham.

[3] Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R. and Srinivasan, P.P., 2021. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. *arXiv preprint arXiv:2103.13415*.

[4] Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B. and Durand, F., 2021. Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering. *arXiv preprint arXiv:2106.02634*.

3D Video from single view video

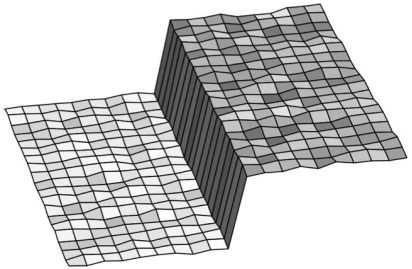
Nupur Kumari, Vivek Roy

Existing work: 3D shot photography

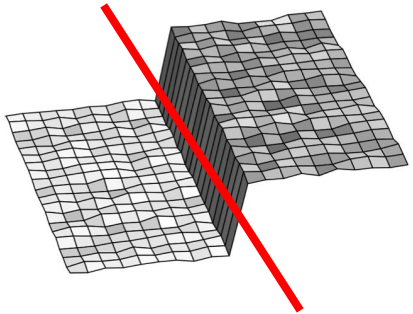


Method flowchart

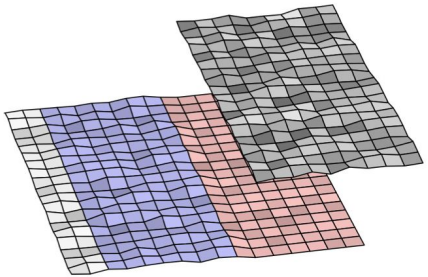
Create a Layered Depth Image (LDI) using depth map



Find discontinuities in LDI by depth threshold

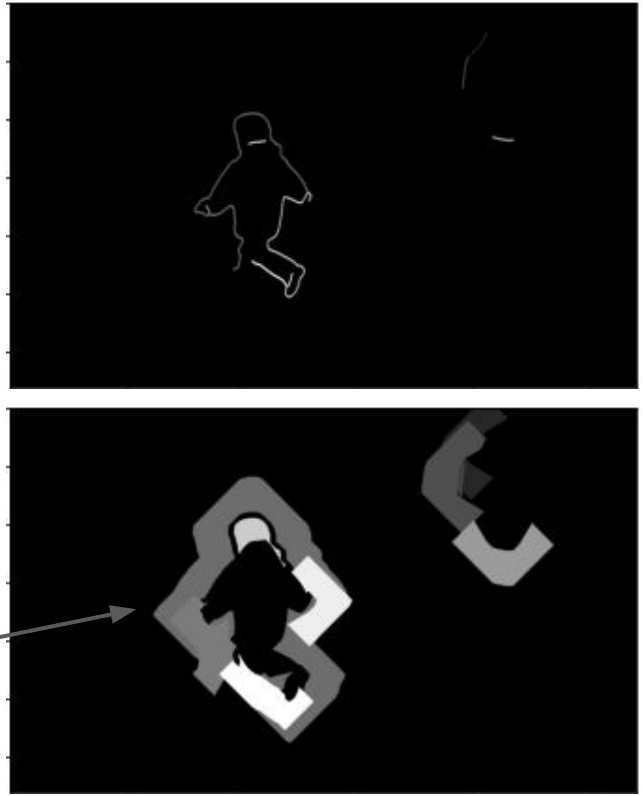


Create context region for in-painting background
(lower depth part)



Example Context and Discontinuity map





Note that each discontinuous edge is handled independently with its own context region



Failure cases

If the context regions extends to wrong regions, in-painting output is not ideal and leads to bleeding.



Failure cases

Similar example, where the road has white patches due to white color in shirt, though its not there in original frame.

Thus the method is quite sensitive to predicted depth values



Simple approach to prevent context bleeding

Use segmentation to remove context from different layers of LDI. (We use detectron2 Mask RCNN)



Before



After



Simple approach to prevent context bleeding

Before



After





Doesn’t solve the issue always. We can use context of other frames in video for in-painting

Before



After



Extended to video frame by frame



Depth estimation from video

Used CVD[*] with updated flow prediction network for improved consistency based depth estimation.

Given camera parameters for each frame map i^{th} depth frame to j^{th} frame:

$$f_{i \to j}(p) = K_j R_j^T (R_i K_i^{-1} f(p) + t_i - t_j)$$

[*] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. ACM TOG (Proc. SIGGRAPH), 39(4), 2020.

Depth estimation from video

Use predicted flow to warp the i^{th} frame to j^{th} frame and then find the 3D coordinate of warped image in C_j coordinate frame.

Measure the similarity between the reprojections using flow and camera.



Flow prediction by RAFT

Teed, Zachary, and Jia Deng. "Raft: Recurrent all-pairs field transforms for optical flow." *European conference on computer vision*. Springer, Cham, 2020..

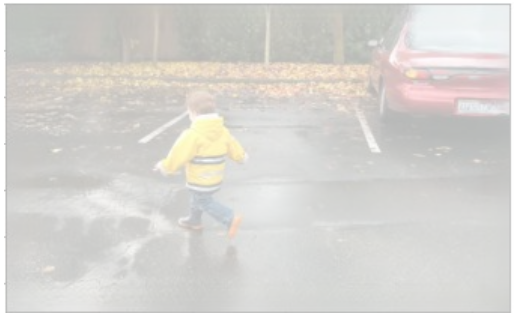


CVD



CVD + RAFT

Using more context from video





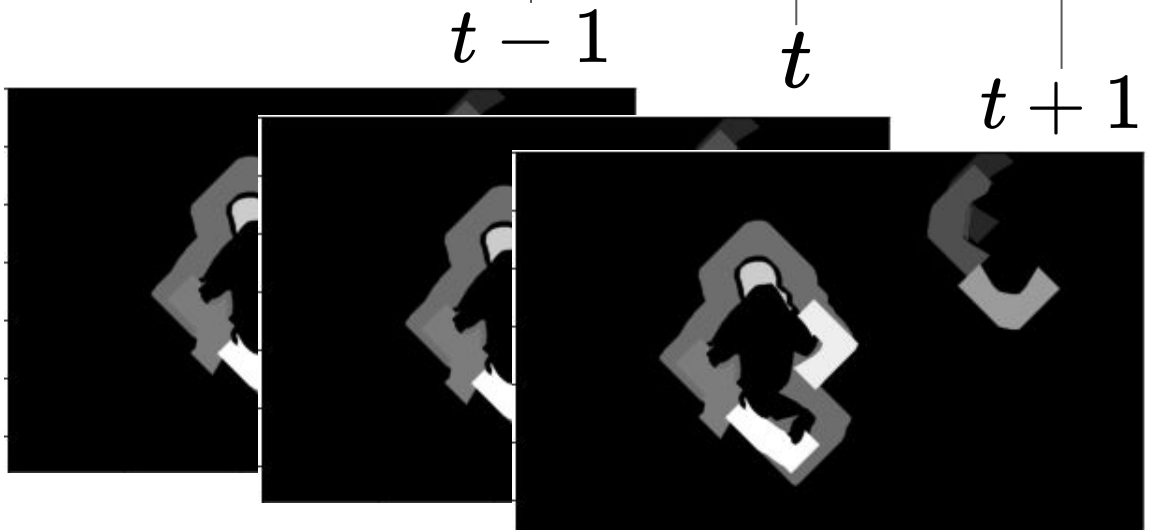


Note that each discontinuous edge is handled independently with its own context region

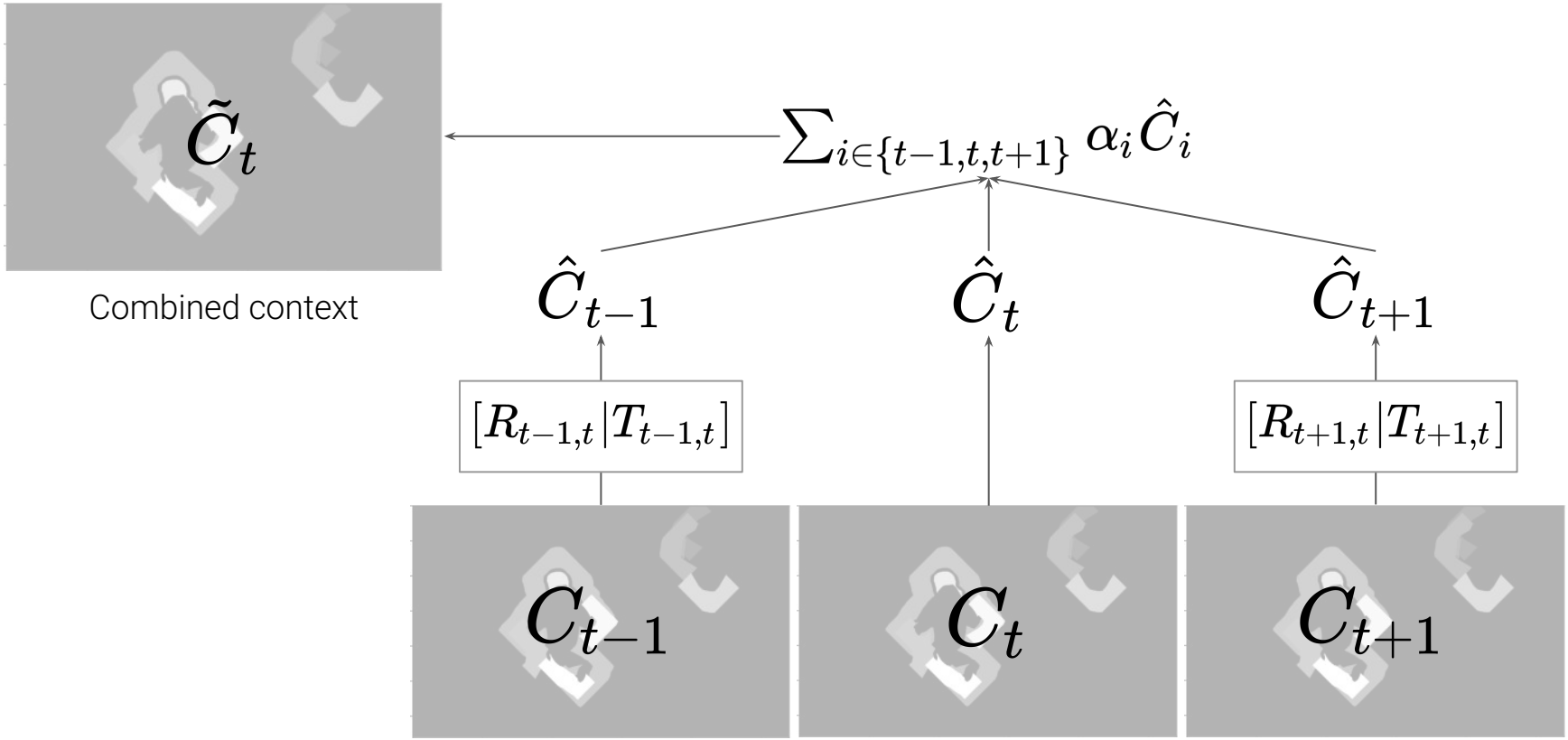
Using more context from video



Combined context



Using more context from video



Using more context from video





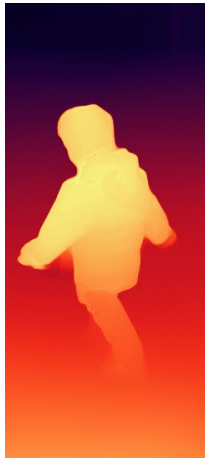
Failure Case



$t = 6$



$t = 5$

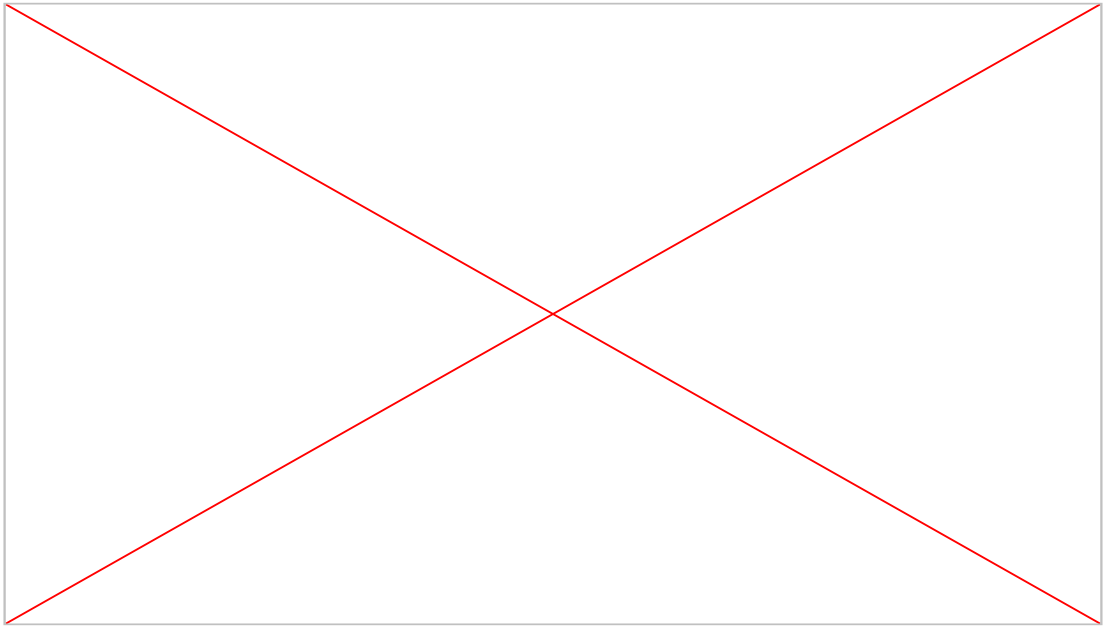


$t = 6$



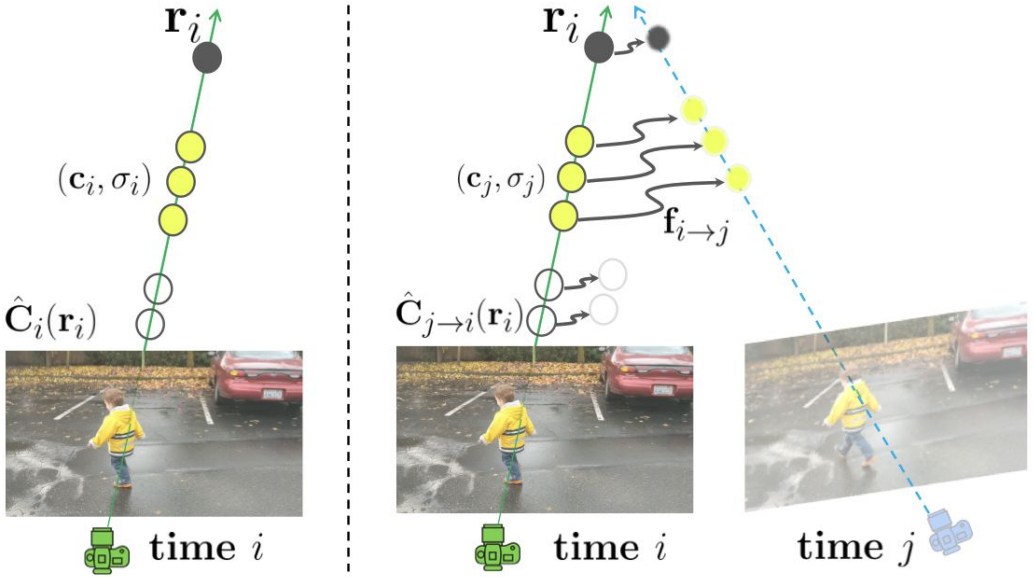
$t = 7$

Temporally consistent Nerf Representation



Nerf video representation

Current training uses flow to train a time consistent NeRF model

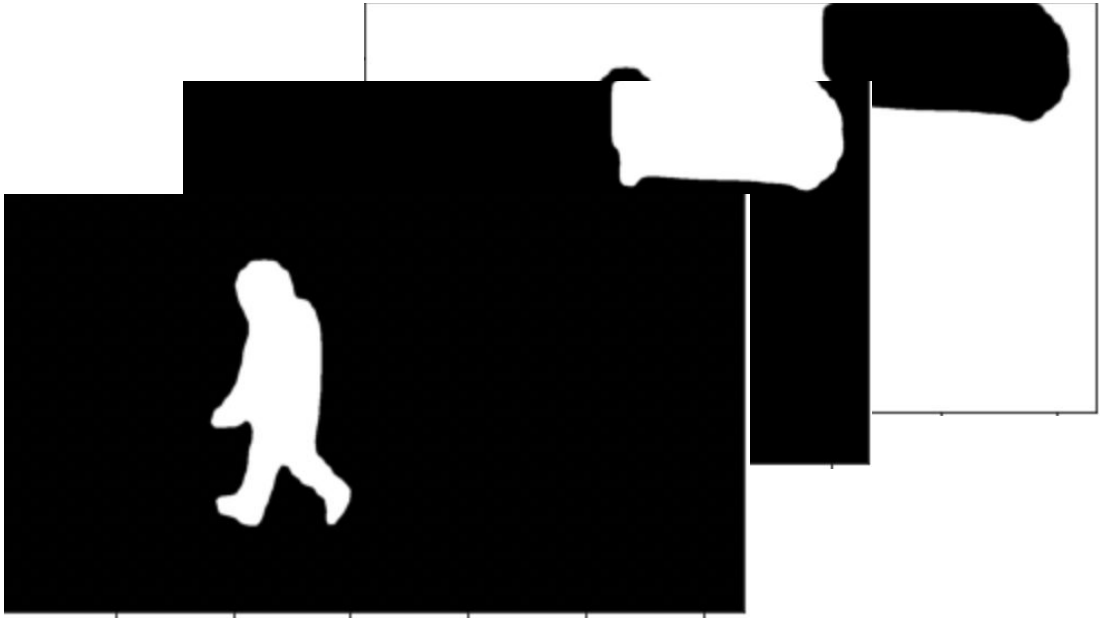


Li, Zhengqi, et al. "Neural scene flow fields for space-time view synthesis of dynamic scenes." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021..

Disentangling objects in Nerf representation

Use different nerf model for each object and fuse them together during rendering.

Promote predicted opacity to match the masks for object branch.



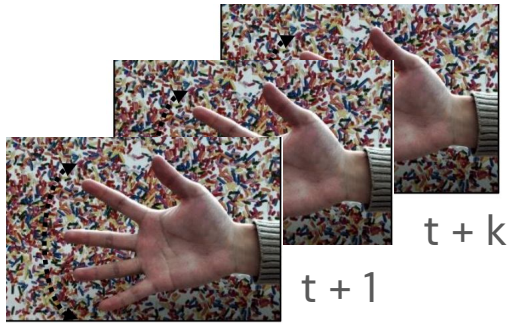
Li, Zhengqi, et al. "Neural scene flow fields for space-time view synthesis of dynamic scenes." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021..

Scene Flow from Light Field Gradients

Joelle Lim

Scene Flow: what is?

Given:



How did the scene move?

+ + +

how...

+ + +



Ray flow equation

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

Ray flow equation

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

Light field gradients

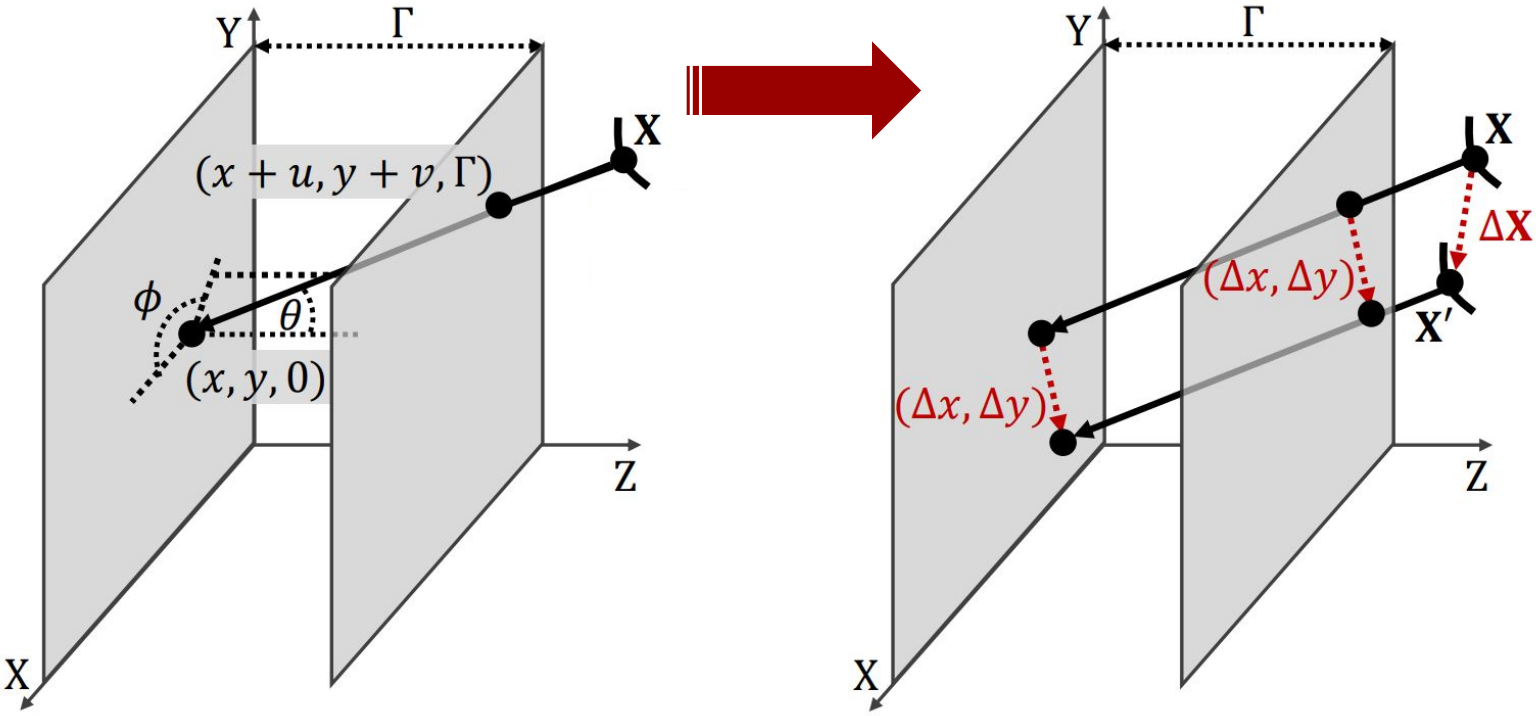
Ray flow equation

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

Light field gradients

Scene Flow Components

parallel shift of ray

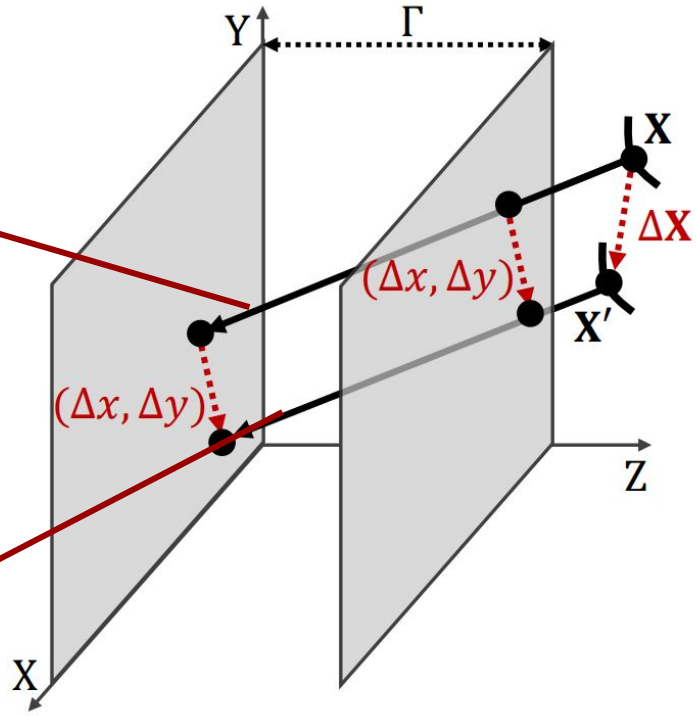


Light ray parameterization

Ray Flow due to scene motion

$L(x, y, u, v, t)$

$L(x + \Delta x, y + \Delta y, u, v, t + \Delta t)$

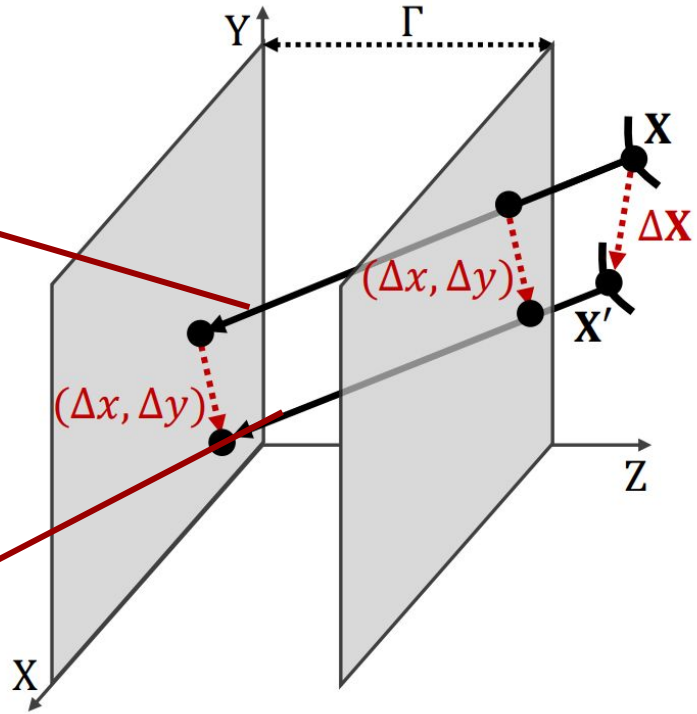


Assuming ray brightness remains constant,...

$$L(x, y, u, v, t)$$

$$=$$

$$L(x + \Delta x, y + \Delta y, u, v, t + \Delta t)$$



$$L(x, y, u, v, t) = L(x + \Delta x, y + \Delta y, u, v, t + \Delta t)$$



First order taylor expansion

$$\frac{\partial L}{\partial x}\Delta x + \frac{\partial L}{\partial y}\Delta y + \frac{\partial L}{\partial t}\Delta t = 0\,.$$

$$L(x,y,u,v,t) = L(x+\Delta x, y+\Delta y, u, v, t+\Delta t)$$

First order taylor expansion

$$\frac{\partial L}{\partial x}\Delta x + \frac{\partial L}{\partial y}\Delta y + \frac{\partial L}{\partial t}\Delta t = 0\,.$$

Substitute

$$L_X = \frac{\partial L}{\partial x},\ L_Y = \frac{\partial L}{\partial y},$$

$$,\ L_Z = -\frac{u}{\Gamma}\frac{\partial L}{\partial x} - \frac{v}{\Gamma}\frac{\partial L}{\partial y},\ L_t = \frac{\partial L}{\partial t}$$

Ray flow equation

$$\boxed{L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0}$$

$$\mathbf{V} = (V_X, V_Y, V_Z) = (\frac{\Delta X}{\Delta t}, \frac{\Delta Y}{\Delta t}, \frac{\Delta Z}{\Delta t})$$

+ + +

Ray flow equation

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

underconstrained

+ + +

+ + +

Ray flow equation

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

underconstrained

+ + +

have to impose additional constraints!!!

Ray flow equation

+ + +

$$\boxed{L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0}$$

Optical flow equation

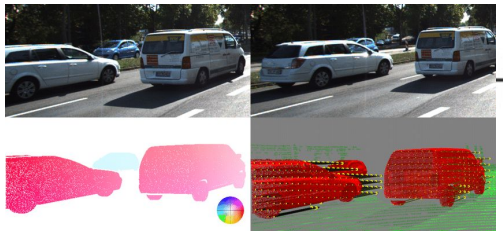
$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} = 0$$

Ray flow equation

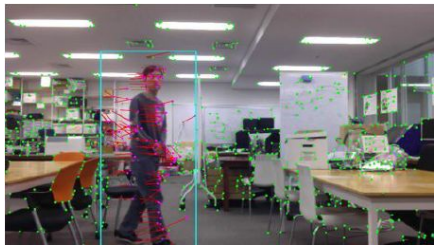
+ + +

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

Optical flow equation







Ray flow equation

+ + +

$$L_X V_X + L_Y V_Y + L_Z V_Z + L_t = 0$$

Optical flow equation

Local:
Lucas Kanade

Global:
Horn Schunck

Lucas Kanade (Local)

assumption: scene motion is constant in neighbourhood

Solve for V, $\mathbf{AV} = \mathbf{b}$

$$\mathbf{A} = \begin{bmatrix} L_X(\mathbf{x}_1) & L_Y(\mathbf{x}_1) & L_Z(\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ L_X(\mathbf{x}_n) & L_Y(\mathbf{x}_n) & L_Z(\mathbf{x}_n) \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -L_t(\mathbf{x}_1) \\ \vdots \\ -L_t(\mathbf{x}_n) \end{bmatrix}$$

where

$$\mathbf{x}_c, \mathbf{x}_i \in \mathcal{N}(\mathbf{x}_c)$$

Horn Schunck (Global)

Optimization problem to minimize
Scene motion variation

Minimize:

$$E(\mathbf{V}) = E_D(\mathbf{V}) + E_S(\mathbf{V}), \qquad \text{where}$$

error term — smoothness term

$$E_D(\mathbf{V}) = \int_{\Omega} (L_X V_X + L_Y V_Y + L_Z V_Z + L_t)^2 \, dx\, dy\, du\, dv \,,$$

$$E_S(\mathbf{V}) = \int_{\Omega} \left(\lambda |\nabla V_X|^2 + \lambda |\nabla V_Y|^2 + \lambda_Z |\nabla V_Z|^2\right) dx\, dy\, du\, dv \,.$$

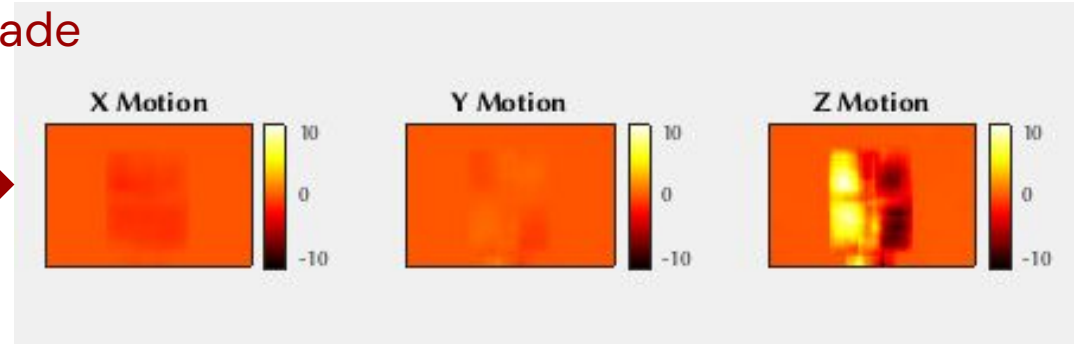
some results

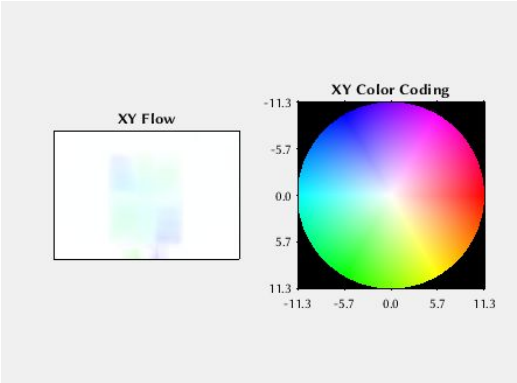
Image

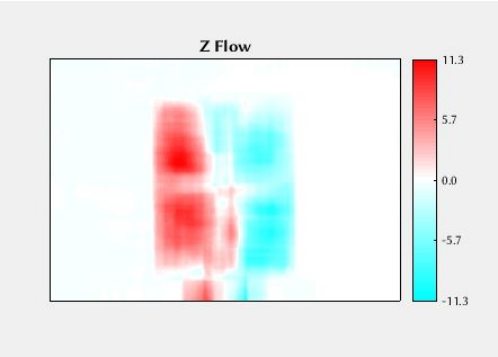


Lucas Kanade









+ + +

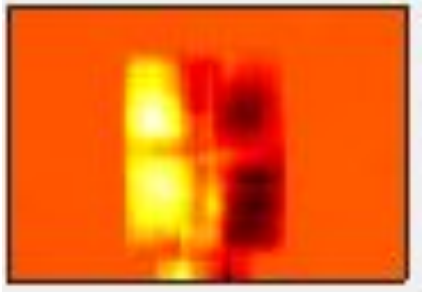
Lucas Kanade

Horn-Schunck

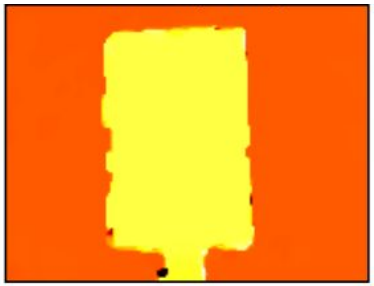
Global + Local

Image









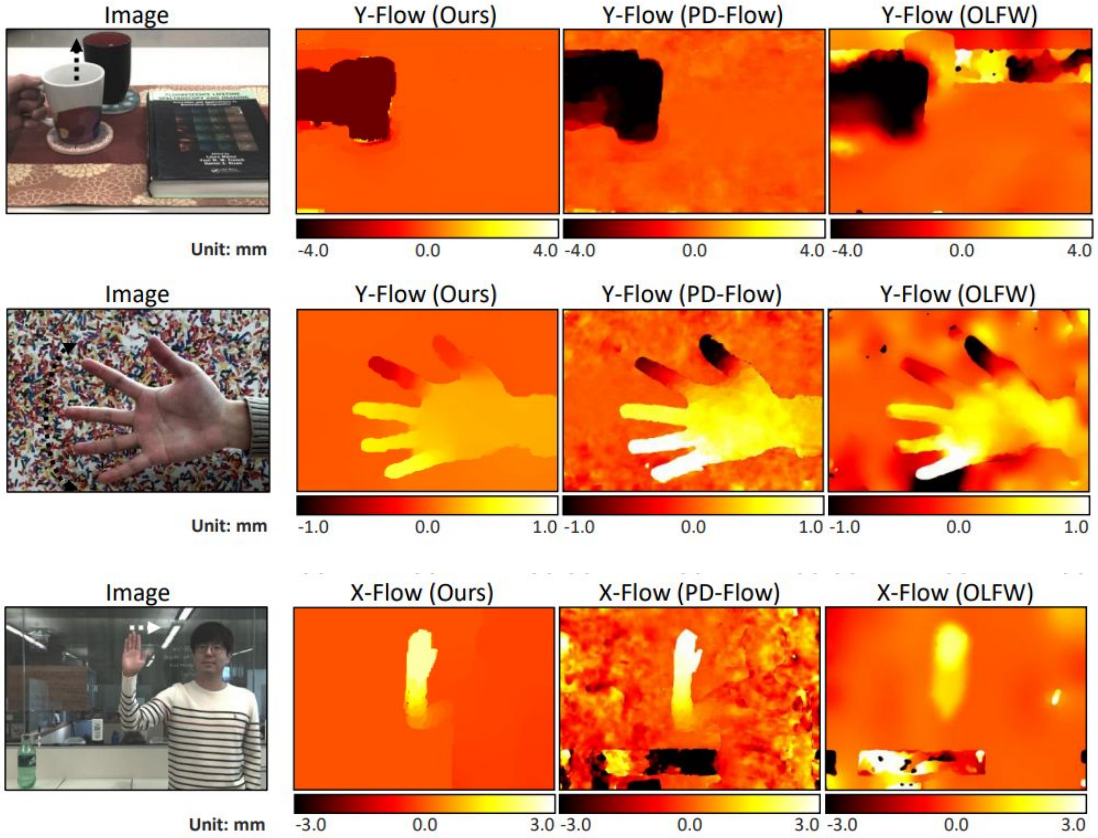
my results

their chad result

Other results from paper

with enhancements ++

- Local + Global methods
- Pyramid
- Graduated Non Convexity iterations
- Occlusions terms



Thank you!

Citations:

Ma, S., Smith, B. M., & Gupta, M. (2018). 3D scene flow from 4d light field gradients. *Computer Vision – ECCV 2018*, 681–698. https://doi.org/10.1007/978-3-030-01237-3_41

Reconstructing Refractive Objects via Light-Path Triangulation

Emma Liu (emmaliu)

Why is reconstruction of transparent objects hard?

Because they don’t have a local appearance - we have to rely on observations of the scene behind the object.

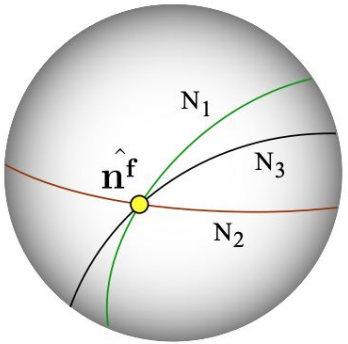
Well that’s difficult. What can we do about it?



Apply knowledge of the scene and properties of refraction to light-path triangulation!

Refractive Light-Path Triangulation

Assuming that light intersects with the surface of an object at most twice (enters and exits), and with light propagation information of *different views of the scene*, solve a minimization problem finitely constraining the surface properties of the object.



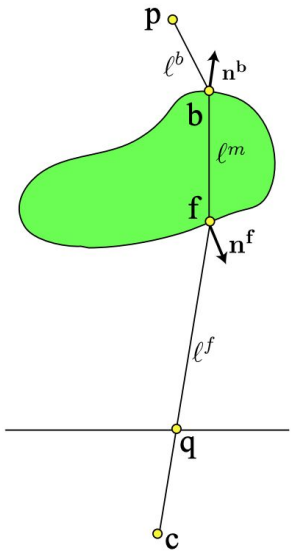
Light Path Scene Model

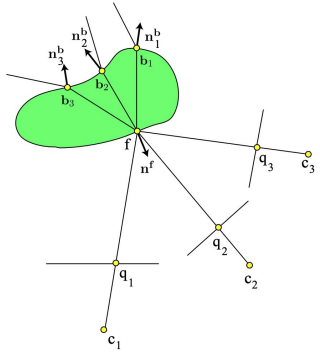
For a pixel **q**, if our view of 3D world **p** is obscured by a transparent object, it takes a piece-linear *light path* and refracts through the object.

Assuming that the light path intersects with the surface exactly 2x, then we can solve for the remainders:

- **b**: 1st point of intersection w/ object
- $\mathbf{l}^{\mathbf{b}}$: ray from **p** to **b**
- $\mathbf{l}^{\mathbf{m}}$: ray connecting $\mathbf{l}^{\mathbf{b}}$, $\mathbf{l}^{\mathbf{f}}$ (refracts incident light in the object's interior)
- **f**: 2nd point of intersection w/ object

With position **f** and its surface normal $\mathbf{n}^{\mathbf{f}}$, we can *compute the depth of the object at that point.*

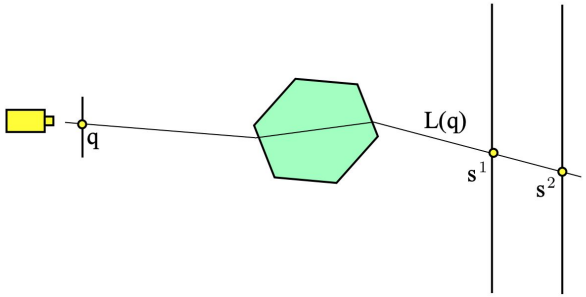




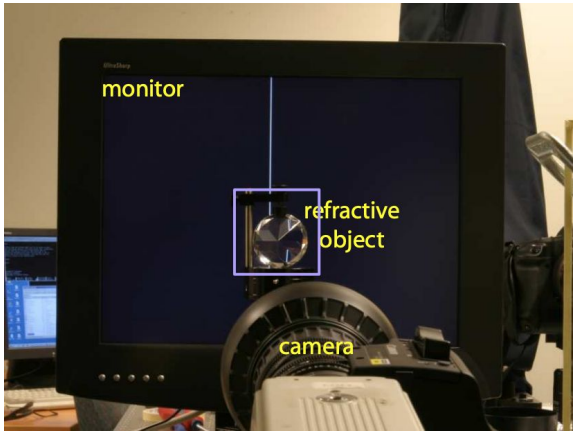
Reference camera vs. with validation ca

Developing the Correspondence Map

For each pixel **q**, determine the first ray in the light path that indirectly projects through it: the first ray that intersects the object, $\mathbf{l}^{\mathbf{b}}$ = L(q).



Key: Make use of two backdrop plane locations to determine two backdrop positions via stripe projection, then map pixel to the ray defined both.



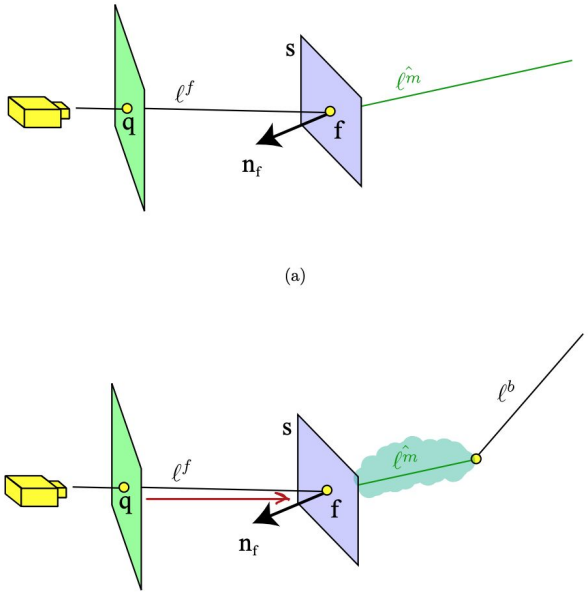
Reconstruction via Minimizing Light-Path Consistency Error

Objective: For each light path, determine the surfel pair (**f**,$\mathbf{n^f}$) that minimizes the reconstruction error: refracted ray $\mathbf{l^m}$ meets $\mathbf{l^f}$ and $\mathbf{l^b}$ at both ends as much as possible.

For each estimate surfel, compute $\mathbf{l^m}$ by tracing backwards and applying Snell's Law with the index of refraction*.

Then the reconstruction error per camera is the shortest distance between the two rays:

$$\mathrm{e}(\hat{\mathbf{s}}) = \mathrm{d}^2(\ell^{\hat{\mathbf{m}}}, \ell^{\mathbf{b}})$$



* But wait, aren't we trying to determine the IOR?? More on that later...

Triangulation Procedure: Depth Sampling

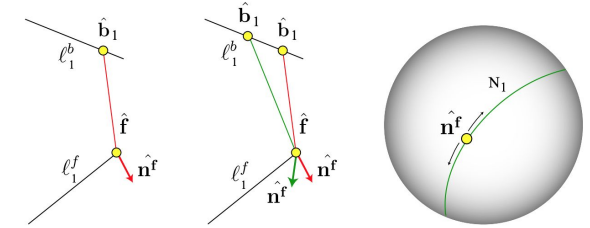
By refraction laws, the normals $\mathbf{n^b}$ and $\mathbf{n^f}$ can be computed based on position **b**, **f**.

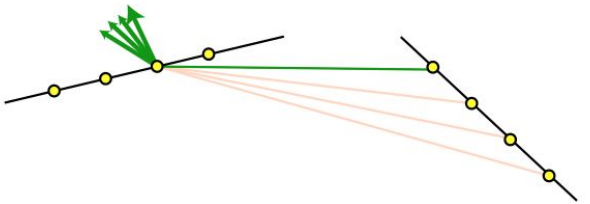
Perform a brute-force search to determine the candidate locations of **b** and **f**:

- Choose finite ranges along rays $\mathbf{l^b}$, $\mathbf{l^f}$.
- Sample n points evenly along this range
- For each sample pair, compute the normal $\mathbf{n^f}$. Evaluate its error.

Depth computed with optimal surfel, camera pos:

$$\hat{\mathbf{f}} = \mathbf{c}_1 + \hat{d}\frac{\ell^{\mathbf{f}}}{\|\ell^{\mathbf{f}}\|}.$$



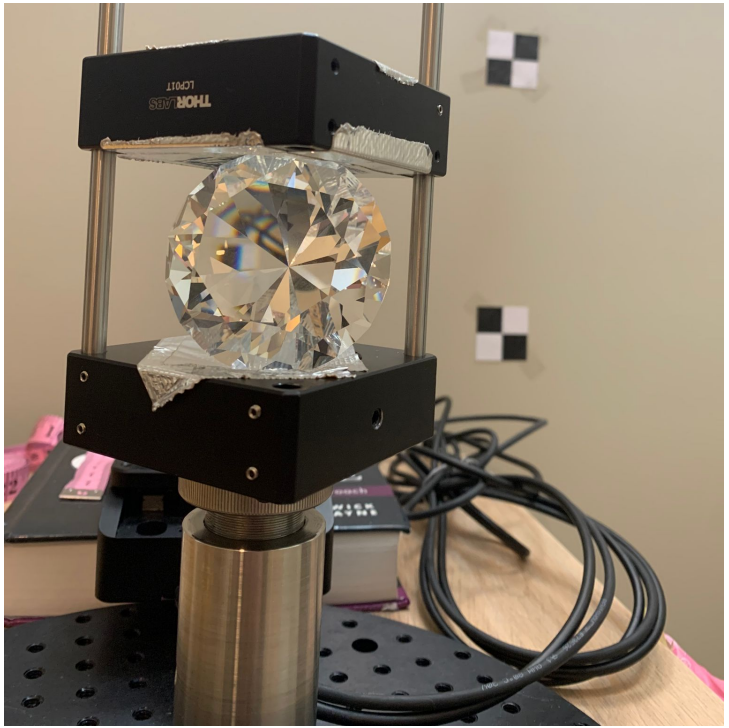


Depth sample along the first and last light-path rays

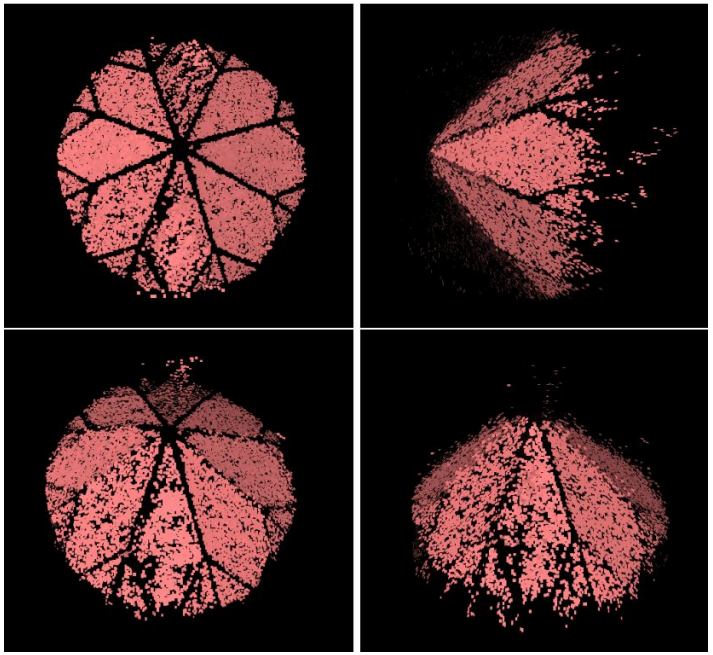
Determining the Index of Refraction (IOR)

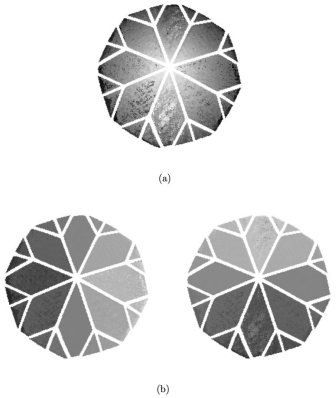
With >=4 cameras, the index of refraction can be computed. Repeat triangulation/optimal surfel-finding procedure for *n* pixels for *m* candidate IORs.

For reference, the IOR of K9 crystal glass (my Bohemian crystal) is ~1.509.

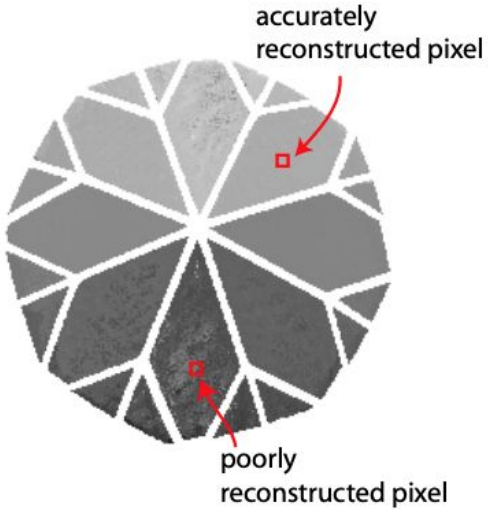


Results





Depth (above), normal maps for rotated views (below)



Thanks

Citations:

K. N. Kutulakos and E. Steger, “A theory of refractive and specular 3D shape by light-path triangulation,” International Journal of Computer Vision, 11-Jul-2007. [Online]. Available: https://link.springer.com/article/10.1007/s11263-007-0049-9.

*Steger, E. (2006). “Reconstructing transparent objects by refractive light-path triangulation”, Master’s thesis, Department of Computer Science, University of Toronto. http://www.cs.toronto.edu/ ~esteger/thesis.pdf.

*Figures and high-level algorithm explanation provided by this paper.

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**,and infographics & images by **Freepik**

Multi-modal 3D reconstruction

Robert Marcus (rbm@)

Background

- Quality and price are typically proportional with LIDAR/TOF sensors
- Useful implementations may be cost prohibitive for lower-end applications
- Sol 3D scanner*
  - $800
  - 1-0.1mm resolution
- Microsoft Kinect**
  - $200 (new, 2014)
  - Single digit mm resolution





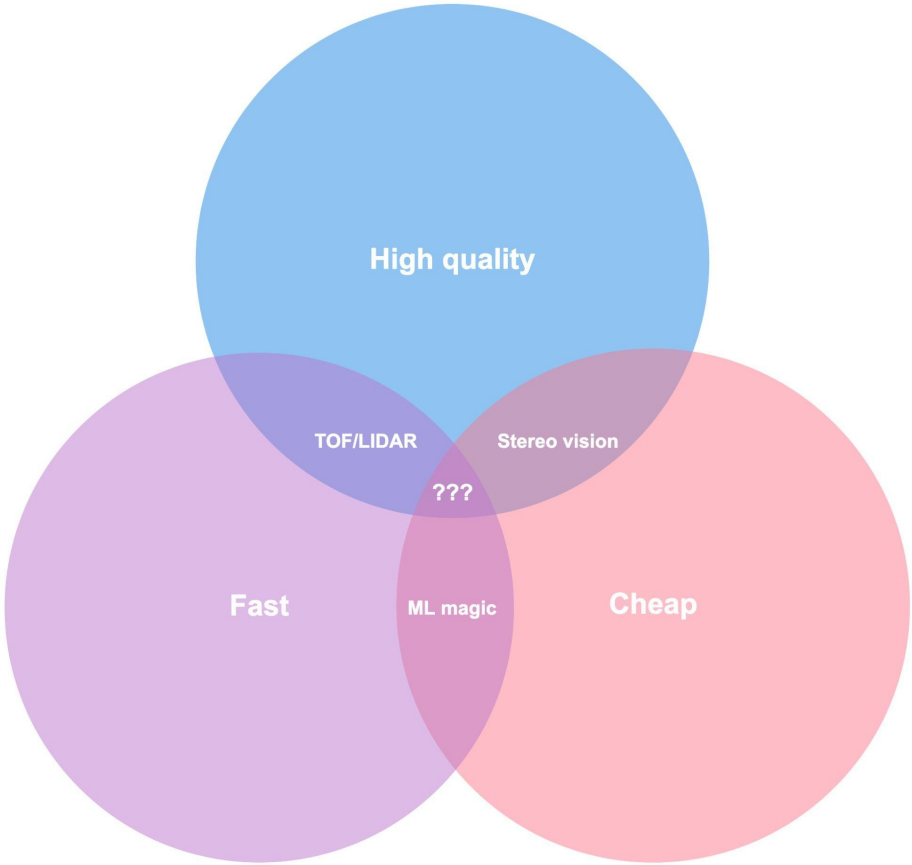
*https://scandimension.com/products/sol-3d-scanner-included-tent-3d-scanner-usb-cables-cover-and-scan-target
**https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3304120/

Background - Cont

- TOF/LIDAR sensors have other shortcomings
  - Size/bulkiness
  - Results can be corrupted (i.e., noisy results depending on lighting)
- Image based alternatives for 3d reconstruction
  - Photometric stereo (Assgn5)
  - Shape from shading
  - Etc.
- Data based alternatives for 3d reconstruction
  - Monocular depth estimation
    - Lot of cool looking monocular stuff here: https://paperswithcode.com/task/depth-estimation

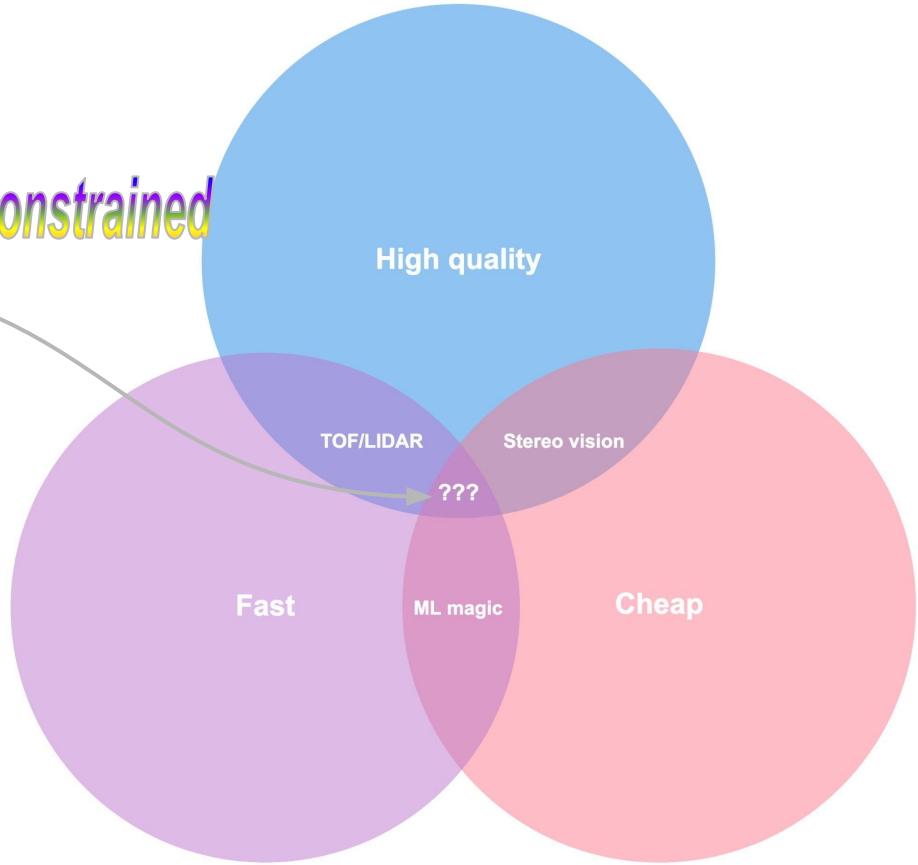
Background - Cont

- Ideal sensor package would have all three properties
- What if we tried ???



Background - Cont

- Ideal sensor package would have all three properties
- What if we tried Using Geometrically Constrained Polarization Normals



GCPNs - Big picture

- Use aligned coarse* depth map to constrain shape (normals) from polarization of scene
- Integrating over the constrained normals will yield a superior result to either method on their own
    - Kadambi et al had >10x resolution boost**



Int J Comput Vis (2017) 125:34–51
DOI 10.1007/s11263-017-1025-7

CrossMark

Depth Sensing Using Geometrically Constrained Polarization Normals

Achuta Kadambi[1] · Vage Taamazyan[1,2,3] · Boxin Shi[1,4] · Ramesh Raskar[1]

Received: 10 May 2016 / Accepted: 31 May 2017 / Published online: 22 June 2017
© Springer Science+Business Media, LLC 2017

Abstract Analyzing the polarimetric properties of reflected light is a potential source of shape information. However, it is well-known that polarimetric information contains fundamental shape ambiguities, leading to an underconstrained problem of recovering 3D geometry. To address this problem, we use additional geometric information, from coarse depth maps, to constrain the shape information from polarization cues. Our main contribution is a framework that combines surface normals from polarization (hereafter polarization normals) with an aligned depth map. The additional geometric constraints are used to mitigate physics-based artifacts, such as azimuthal ambiguity, refractive distortion and fronto-parallel signal degradation. We believe our work may have practical implications for optical engineering, demonstrating a new option for state-of-the-art 3D reconstruction.

Communicated by Rene Vidal, Katsushi Ikeuchi, Josef Sivic and Christoph Schnoerr.

✉ Achuta Kadambi
achoo@mit.edu

✉ Boxin Shi
shiboxin@media.mit.edu; boxin.shi@aist.go.jp

Vage Taamazyan
vaheta@gmail.com; vaheta@tardis3d.ru

Ramesh Raskar
raskar@media.mit.edu

1 MIT Media Lab, 75 Amherst Street, Cambridge, MA 02139, USA

2 Skoltech-MIT Initiative, Moscow Institute of Physics and Technology, Moscow, Russia

3 Tardis 3D Technologies LLC, Moscow, Russia

4 Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

Keywords Computational photography · Light transport · Depth sensing · Shape from polarization

1 Introduction

Today, photographers use polarizing filters on 2D cameras to create aesthetic photographs. But what if a polarizing filter is used in the context of 3D photography? In the vein of computational imaging, we present a co-design of polarized optics and post-capture processing for the task of 3D imaging.

Currently, consumer 3D cameras like the Microsoft Kinect produce depth maps that are often noisy and lack sufficient detail. Using computational processing alone is unlikely to mitigate the problem (e.g., if the noise is filtered, so is the detail). As such, there is recent interest in using a *joint* optical and computational approach to both enhance detail and remove noise. One of the most promising solutions is to combine the captured, coarse depth map with surface normals obtained from photometric stereo (PS) or shape-from-shading (SfS). This depth-normal fusion is logical—the coarse depth map provides the geometric structure and the surface normals capture fine detail to be fused. Encouraging results have been shown by several papers that combine low-quality depth maps with surface normal maps obtained from SfS or PS.[1] Unfortunately, SfS and PS are limited by similar scene assumptions, e.g., restrictive lighting and material assumptions. As a complementary technique, this paper proposes the first use of surface normals from polarization to enhance depth maps. While our proposed technique also has assumptions on lighting and material properties, these

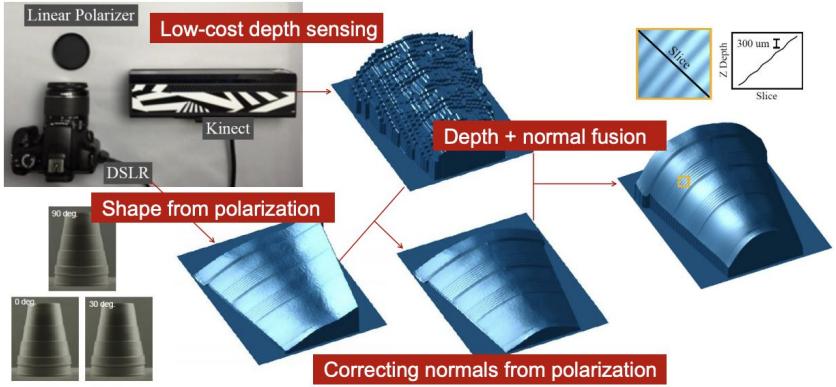
[1] For example, (Yu et al. 2013; Han et al. 2013; Wu et al. 2014) using SfS, and (Nehab et al. 2005; Haque et al. 2014) using PS.

Springer

*1-5mm resolution
**5mm kinect resolution -> .3mm combined resolution

GCPNs - How

- Very complicated. TLDR:
- Normals from polarization by Fresnel equations
- Crutch of the problem comes down to solving several separate systems
  - I.e., solve for Z to get improved normals
  - And more (physics based integration)
- Significant additional work done to reduce ambiguities and distortions by physics based integration (see far right result vs raw fusion)



$$\begin{bmatrix} \lambda\mu I \\ \dots \\ N \cdot T \end{bmatrix} \begin{bmatrix} Z \end{bmatrix} = \begin{bmatrix} \lambda\mu Z^m \\ \dots\dots \\ 0 \end{bmatrix}$$

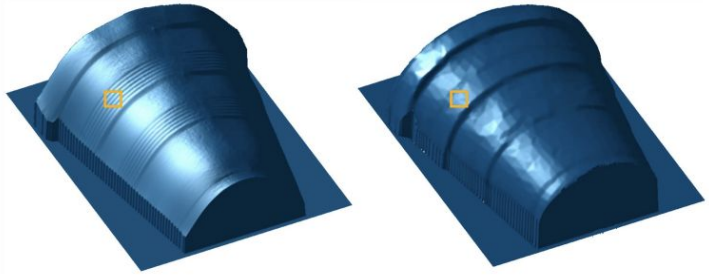
Image credit: https://web.media.mit.edu/~achoo/polar3D/polarized3D_poster.pdf

GCPNs - Results

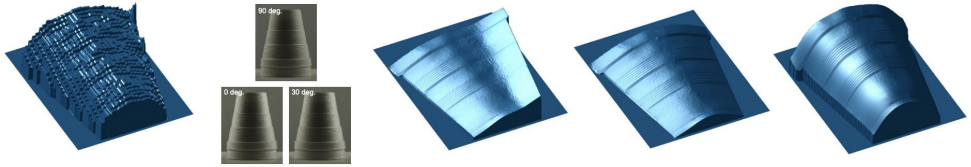
- Physics based poisson integration of fused data yielded superior results
  - Allows correcting refractive distortion and distortion of zenith angle
  - At a cost though…

❑ Comparison with HQ Laser Scan





(a) Enhanced Kinect Depth **(b)** Laser Scanner Depth



(a) Input: Kinect Only
Microsoft Kinect Version II

(b) Input: Polarization Photos
Canon T3i DSLR
Hoya CIR-PL Filter

(c) Naive Shape from Polar.
Shape from Polarization

(d) Our Intermediate Result
Coarse depth to correct azimuthal ambiguity artifacts

(e) Our Final Result
Correcting refractive distortion and physics-based integration

Table 1. Profiling key steps of the algorithm

Step	Runtime
Converting DSLR Photos	59.3 seconds
Plane Principal Component Analysis	56.1 seconds
Azimuth correction	11 minutes, 46 seconds
Physics-based Integration	72 minutes, 24 seconds

Image 1 credit https://web.media.mit.edu/~achoo/polar3D/polarized3D_poster.pdf
Image 2 credit https://web.media.mit.edu/~achoo/polar3D/kadambi_new_england_vision2015.pdf
Table credit https://web.media.mit.edu/~achoo/polar3D/camready/supplement_iccv.pdf

GCPN - My results

- Work in progress
- Using additional (simpler) papers to guide process
- Namely
  - Efficiently Combining Positions and Normals for Precise 3D Geometry (Nehab et al)
    - For sensor fusion
  - Depth from a polarisation + RGB stereo pair (Zhu et al)
    - For integration of the fused data

Credits

- Special thanks to Prof. Yannis for helping me read through the core paper and suggest additional literature
- Polarized 3D: High-Quality Depth Sensing with Polarization Cues
  - Achuta Kadambi and Vage Taamazyan and Boxin Shi and Ramesh Raskar
- Efficiently Combining Positions and Normals for Precise 3D Geometry
  - Diego Nehab and Szymon Rusinkiewicz and James Davis and Ravi Ramamoorthi
- Depth from a polarisation + RGB stereo pair
  - Dizhong Zhu and William A. P. Smith

- Thank you for listening

Synthetic Depth-of-Field with a Mobile Phone

Arpita Nag

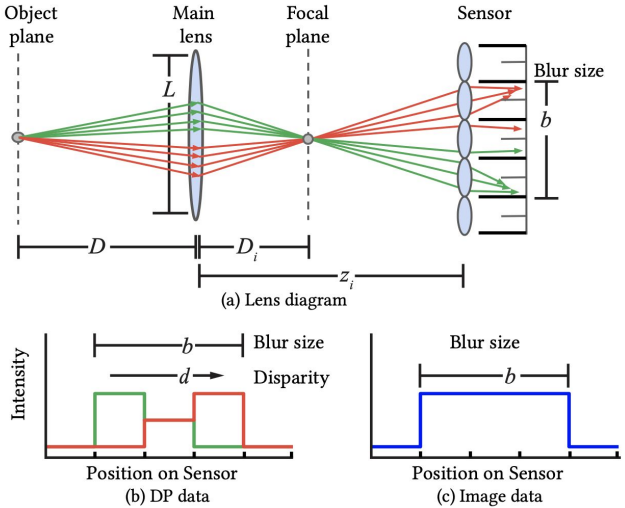
Motivation and Background

- A shallow depth of field and resulting *bokeh* effect is an aesthetic quality in photos
- Difficult to produce on typical smartphone cameras with small apertures, and DSLR's are pretty expensive on their own.
- DP data can serve as a 2D lightfield, or a stereo system with a tiny baseline -- maybe use this disparity to mimic bokeh!



Pipeline

Part 1 - Obtain a Defocus Map



First, we must estimate a high resolution, sub-pixel disparity map.

We first use tile matching of one view against the other, and heuristics to produce a per-pixel flow field (to the other view) and per-pixel confidence.

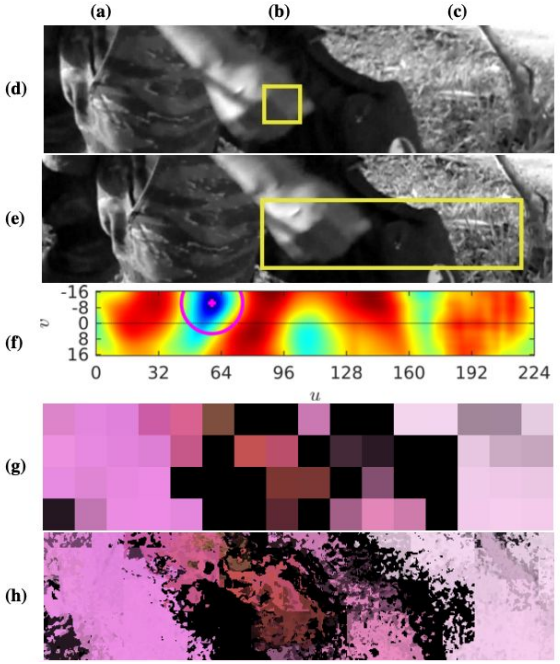


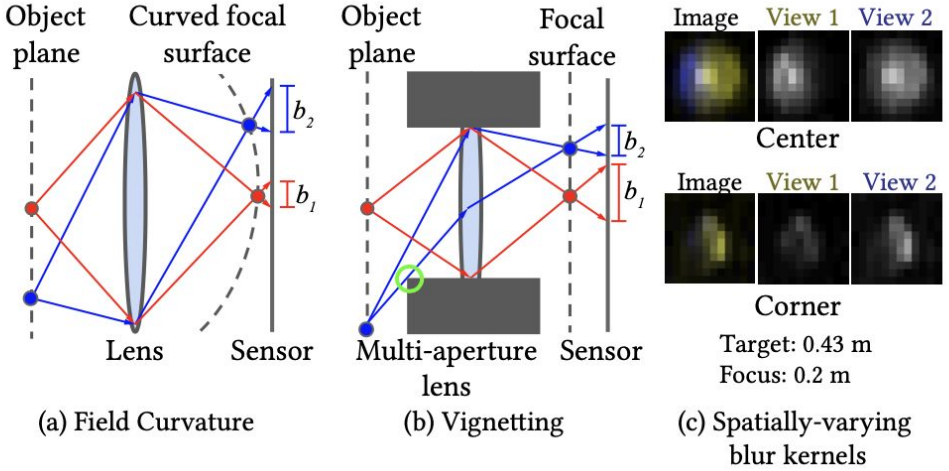
Figure: Original paper which used 2D tile-matching to estimate flow in *temporal sequence.*

Pipeline (Cont)

Part 1.2 - Calibrate the Image

$$d_{corrected}(\mathbf{x}) = I_z(\mathbf{0}) + \frac{S_z(\mathbf{0})(d(\mathbf{x}) - I_z(\mathbf{x}))}{S_z(\mathbf{x})}.$$

Sadly, smartphone cameras don't totally obey the thin-lens approximation.



- Objects at the same depth but different spatial locations can have different disparities due to lens aberrations and sensor defects.

Pipeline (Cont)



Part 1.3 - Final Depth / Defocus Map

- Use bilateral filtering methods and confidence map to produce a smoothed out, in-painted disparity map.

- SIDE NOTE: The original paper also uses trained convolutional neural networks to semantically segment out “people” if they appear in any pics and mask out the region to constant disparity.

Pipeline (End)

Part 2 - Rendering the Blur

$$r(\mathbf{x}) = \kappa(z)\max\left(0, \left|d(\mathbf{x}) - d_{focus}\right| - d_{\emptyset}\right)$$

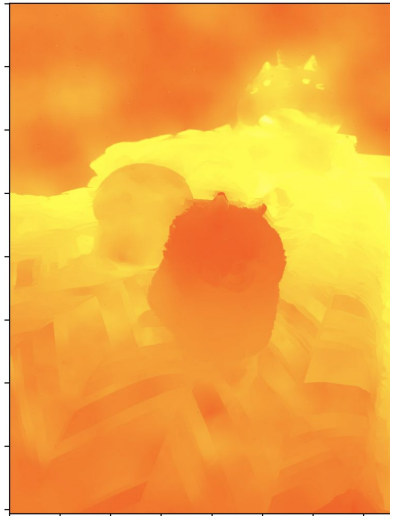
$$B_{scatter}(\mathbf{x}) = \sum_{\Delta\mathbf{x}} I(\mathbf{x}+\Delta\mathbf{x})K_{\mathbf{x}+\Delta\mathbf{x}}(-\Delta\mathbf{x})$$

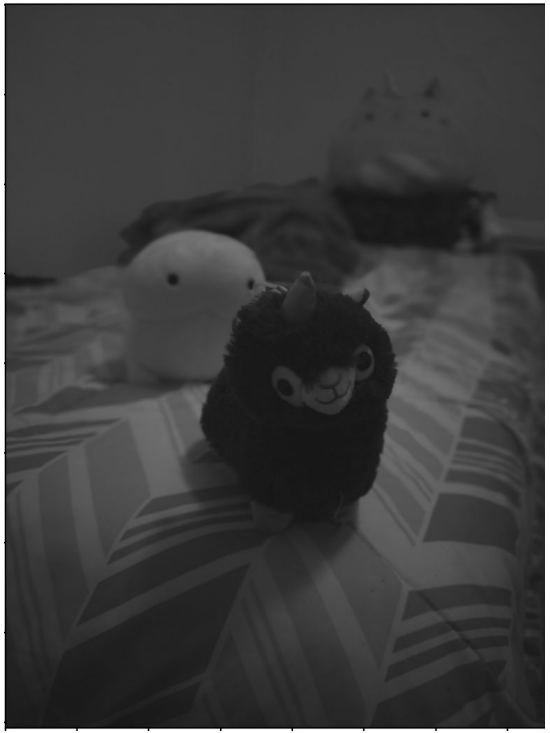
$$\alpha_j = \left(1 + \frac{1}{\eta}\min(d - d_{j-1}, d_j - d)\right)\Big|_{[0,1]}$$

- Precompute a blur radius based on the difference between disparity at location “x” and the target in-focus disparity.
- With optimizations for time, perform a “scatter” operation on the image (similar to convolution)
- Once “blurring” is done for some disparity ranges, use alpha compositing to render final image

Results (Pt 1)





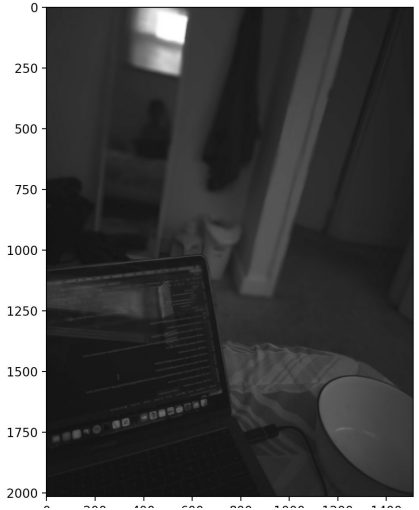


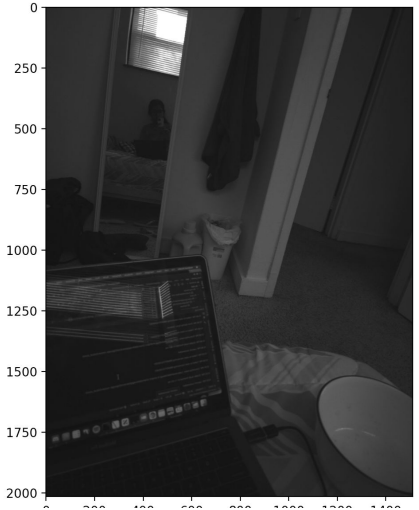
Disclaimer: Calibration not done yet, and rendering is still done with discrete, quantized “disparity bands”.

Results (Pt 2)


0
200
400
600
800
1000
1200
1400


0
250
500
750
1000
1250
1500
1750
2000
0
200
400
600
800
1000
1200
1400

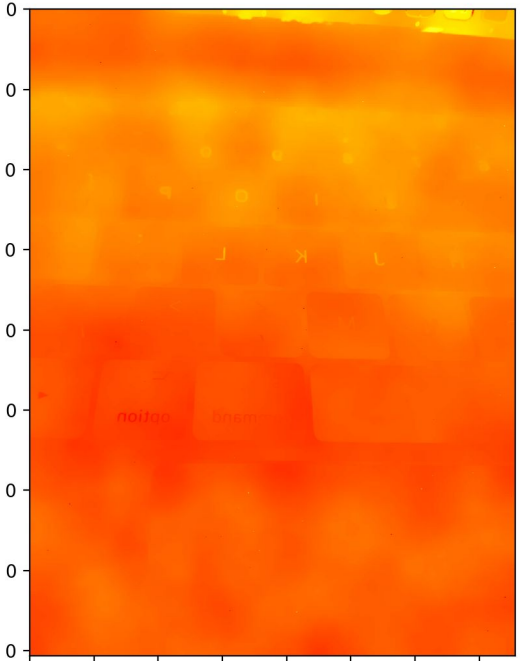

0
250
500
750
1000
1250
1500
1750
2000

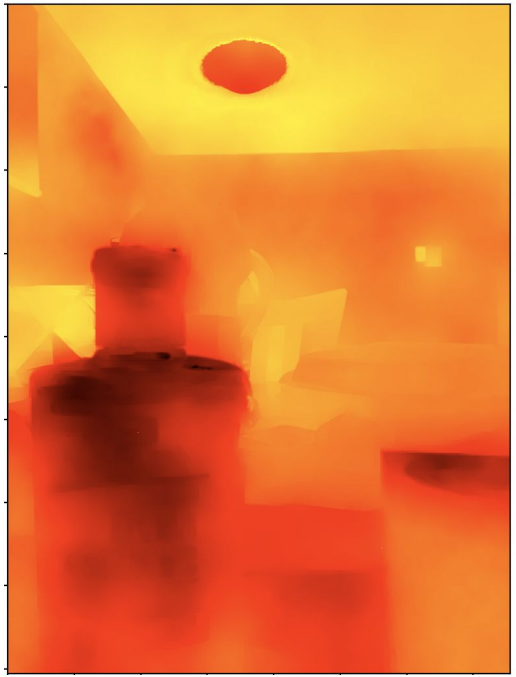

0
250
500
750
1000
1250
1500
1750
2000





More Results and Challenges





- Current rendering implementation is still painfully slow
  - Solution? Blur pixels in the gradient domain or perform blur on downsampled image
- Initial data is still too noisy, even after having captured a burst of images
- Blurred result is unrealistically smooth, so we need to synthetically add noise as in the paper

References

1. **Robert Anderson, David Gallup, Jonathan T. Barron, Janne Kontkanen, Noah Snavely, Carlos Hernández, Sameer Agarwal, and Steven M. Seitz. 2016. Jump: virtual reality video. ACM Trans. Graph. 35, 6, Article 198 (November 2016), 13 pages. DOI:https://doi.org/10.1145/2980179.2980257**

2. **Barron J.T., Poole B. (2016) The Fast Bilateral Solver. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9907. Springer, Cham. https://doi.org/10.1007/978-3-319-46487-9_38**

3. **Neal Wadhwa, Rahul Garg, David E. Jacobs, Bryan E. Feldman, Nori Kanazawa, Robert Carroll, Yair Movshovitz-Attias, Jonathan T. Barron, Yael Pritch, and Marc Levoy. 2018. Synthetic depth-of-field with a single-camera mobile phone. ACM Trans. Graph. 37, 4, Article 64 (August 2018), 13 pages. DOI:https://doi.org/10.1145/3197517.3201329**

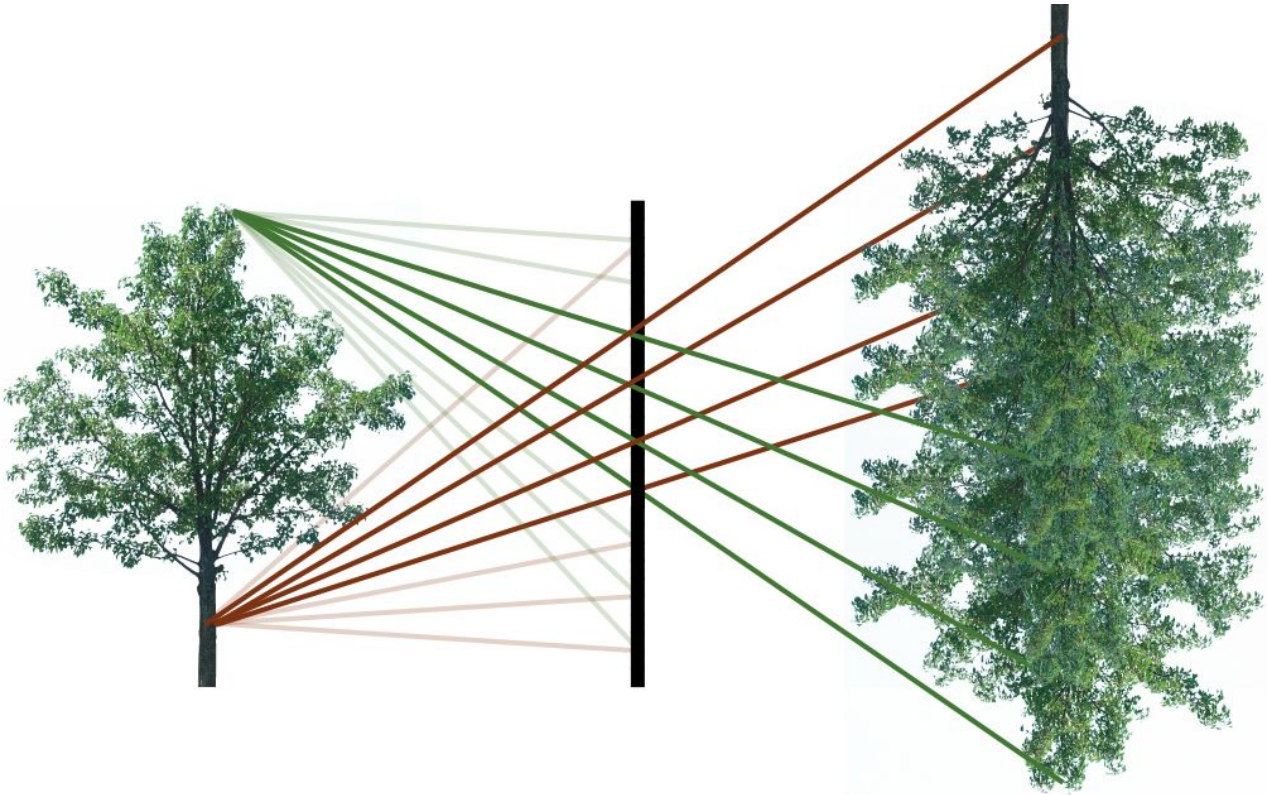
Thank you!

Any questions?

Capturing Lightfields

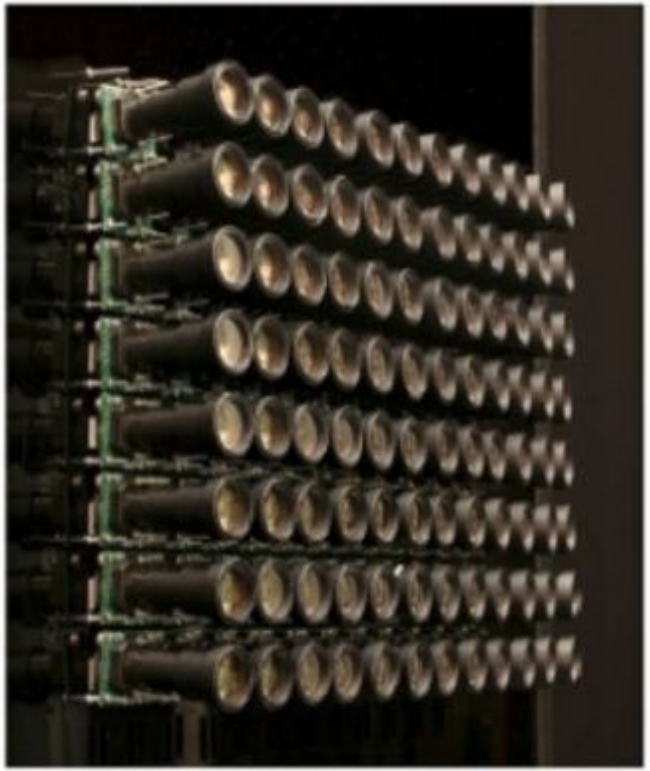
Gaurav Parmar

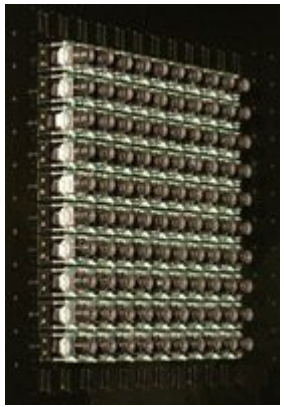
Task Goal

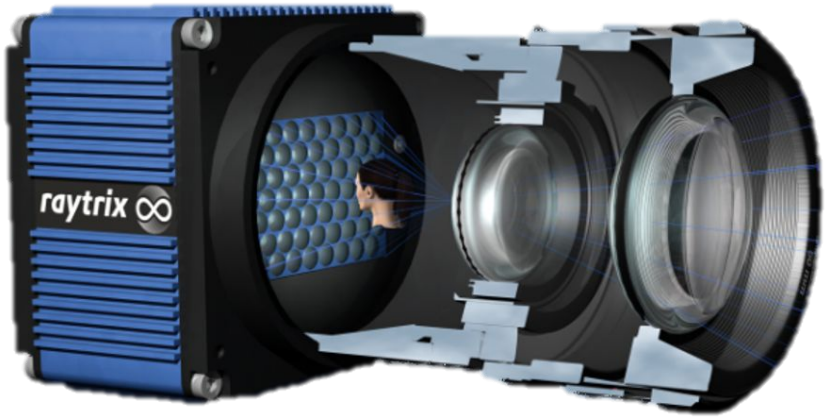


[1]

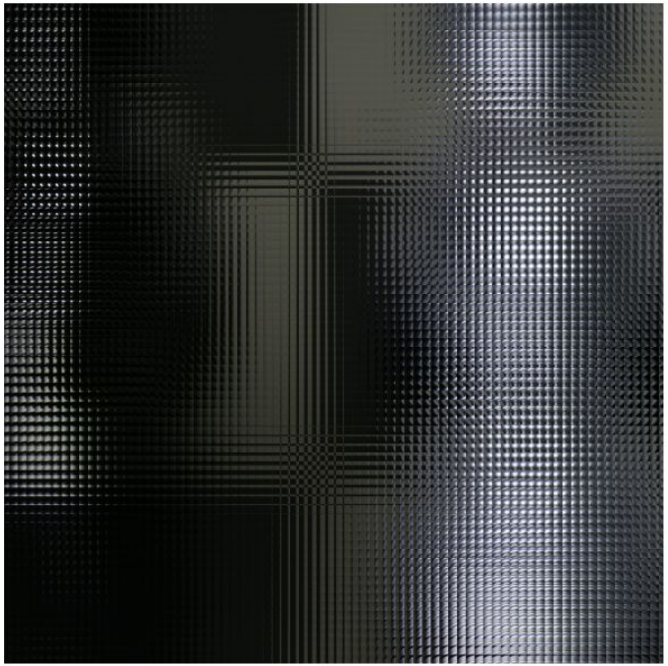
Hardware for Capturing the Lightfields







Example lightfields





Refocusing the Image





Aspects of this approach

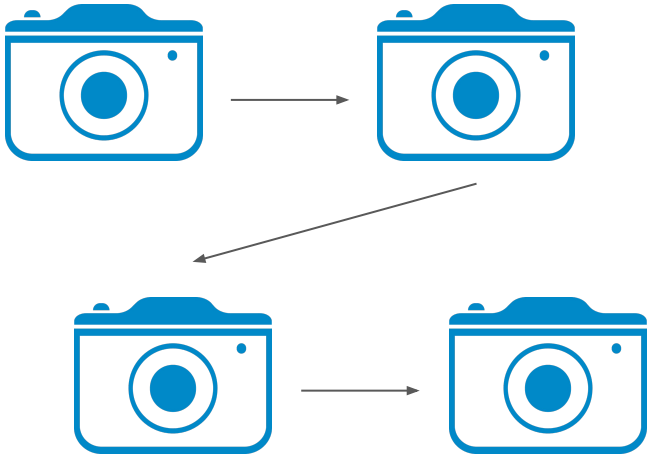
(+) the lighfields captured are great

(-) very expensive equipment is needed

- Can ***I*** capture something like this?

Unstructured Lightfields

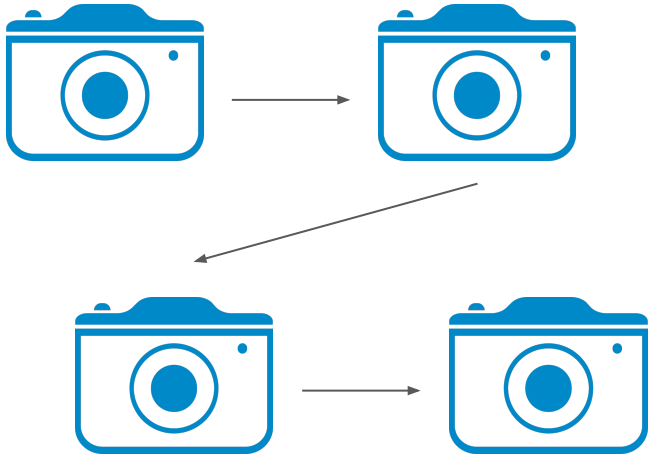
- Can we capture the lighfield by moving the camera around?
- Aligned later using a template matching method





Challenges in this Procedure

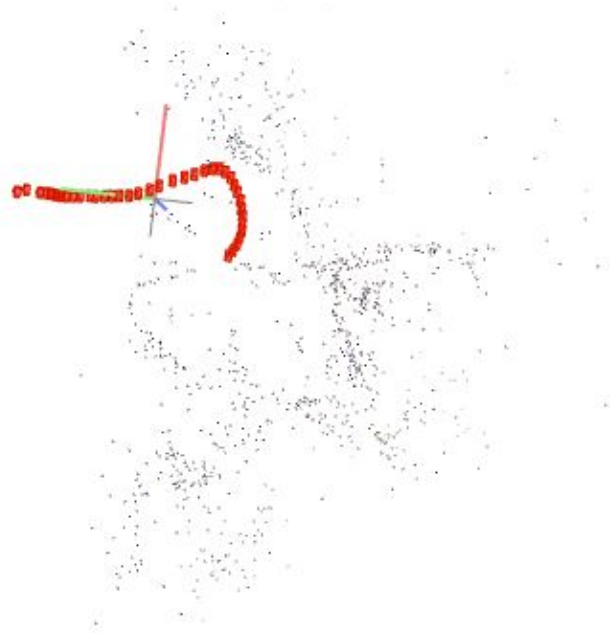
- The motion of the camera needs to be carefully planned



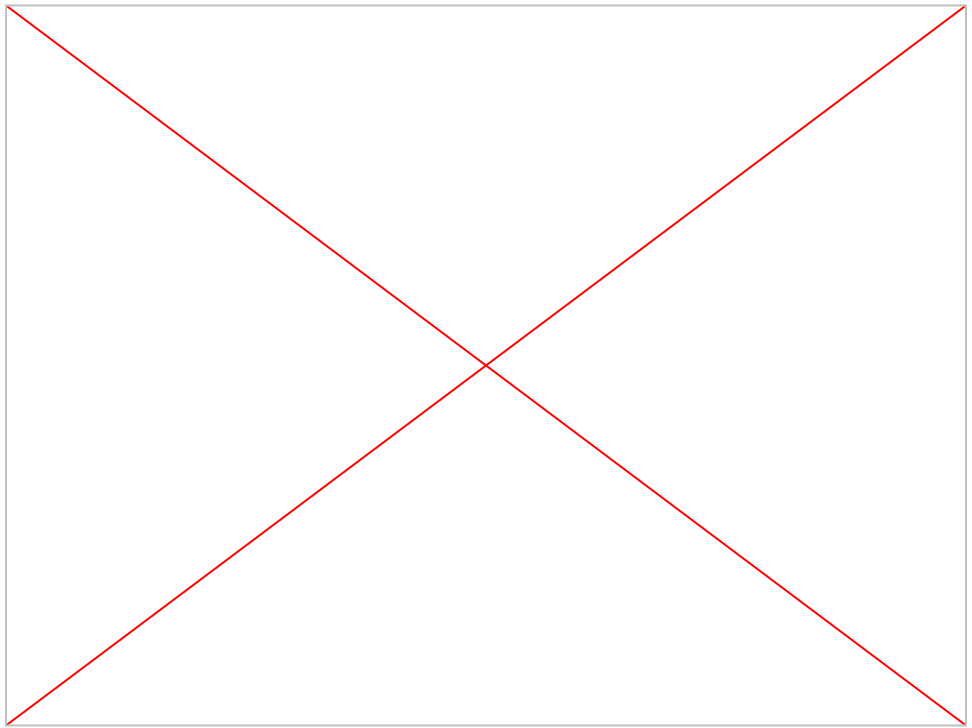


Solution - filtering out frames

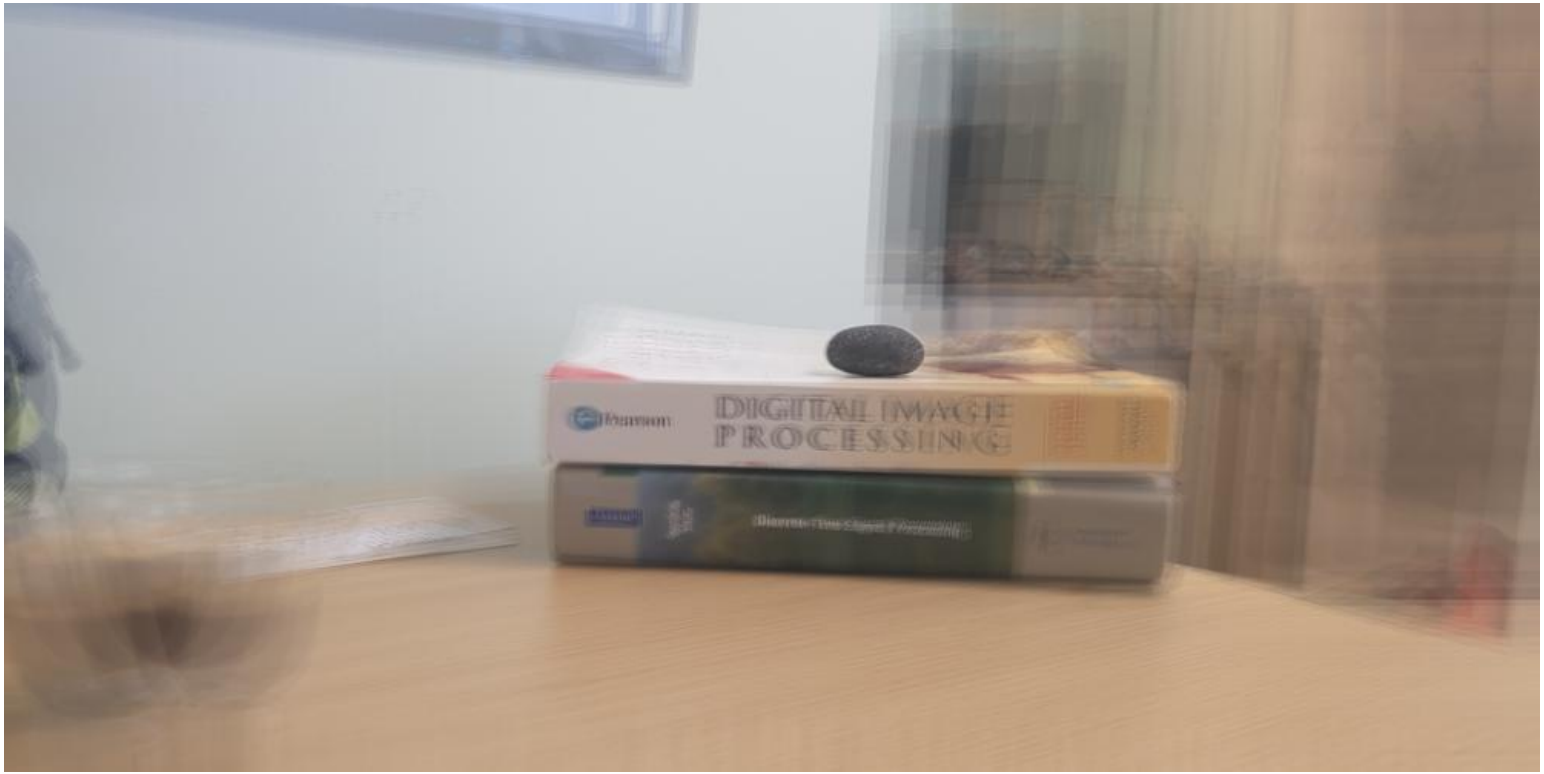
- Track the position of each frame in the video



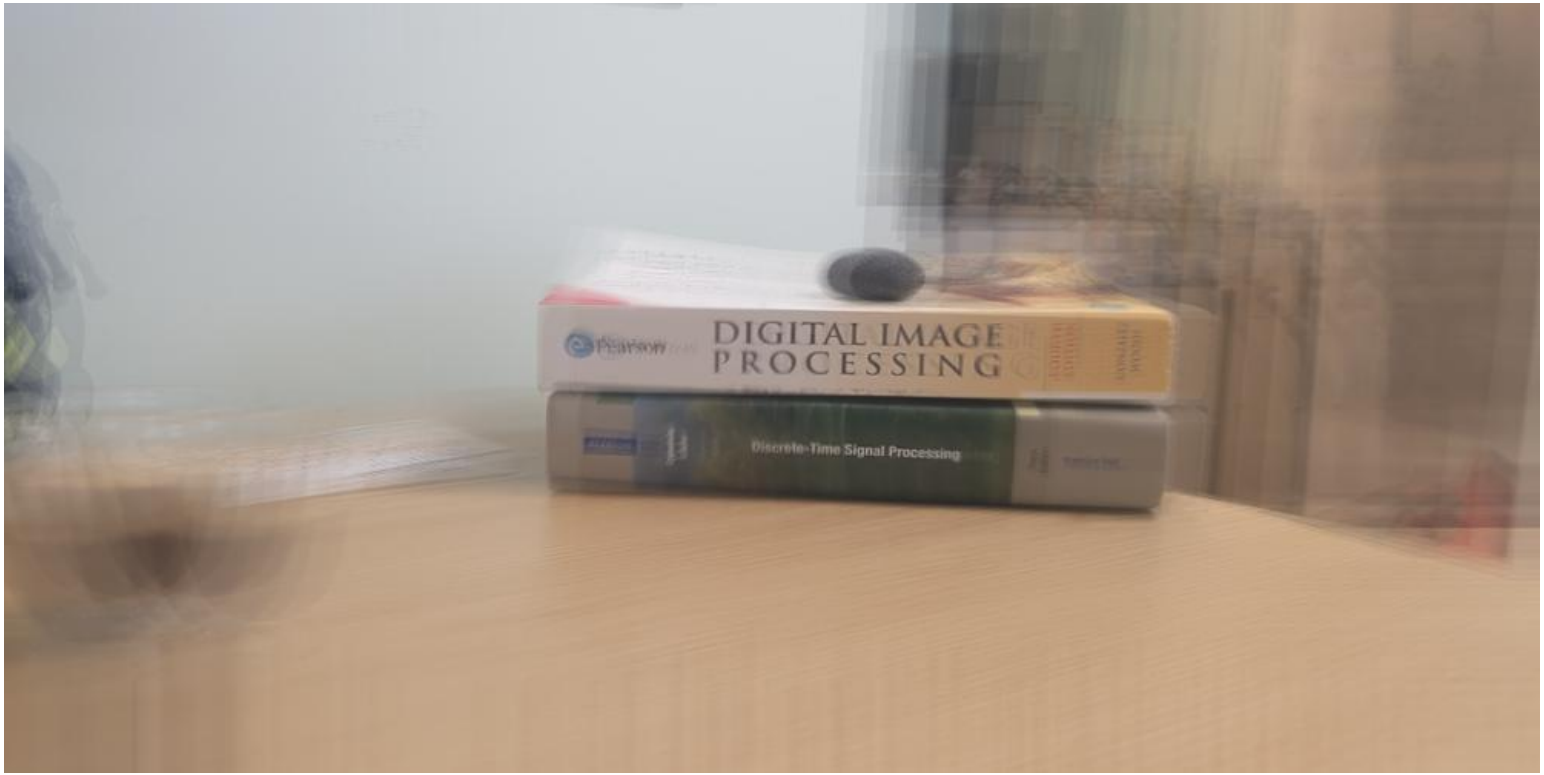
Initial Result Captured



Focus on the stone

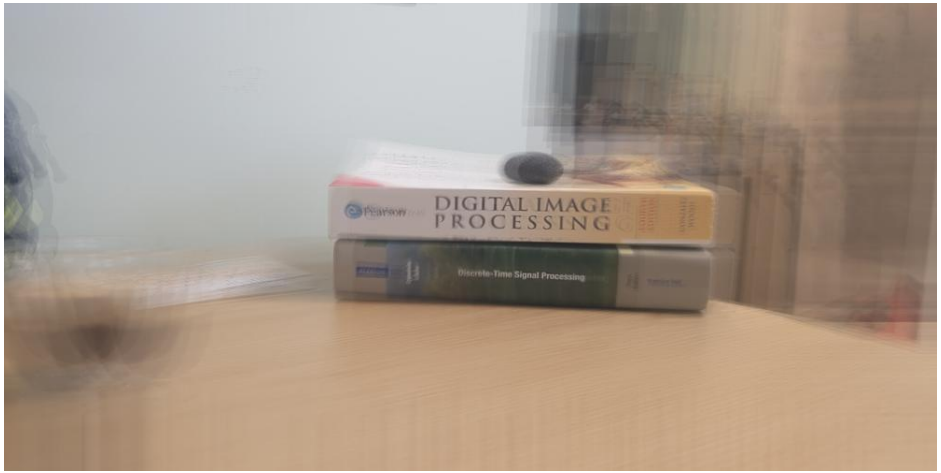


Focus on the Text

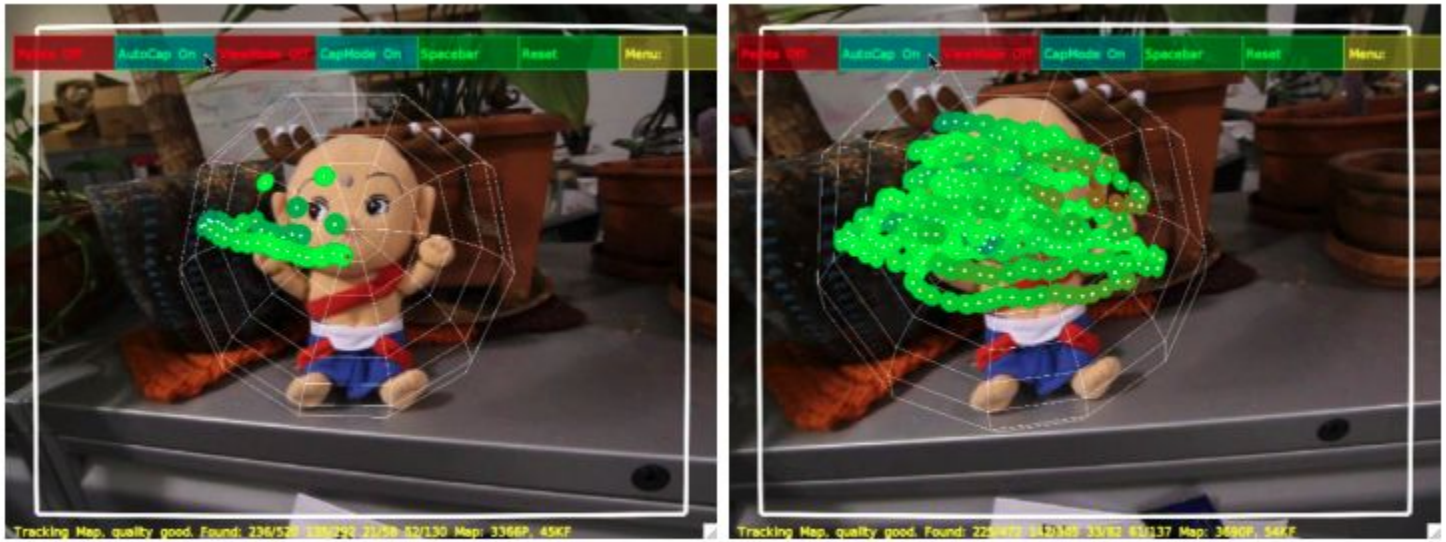


Issues in the results

- The results captured still do not look great
- Interface to close the loop



In progress tasks



[2]

Thank You!

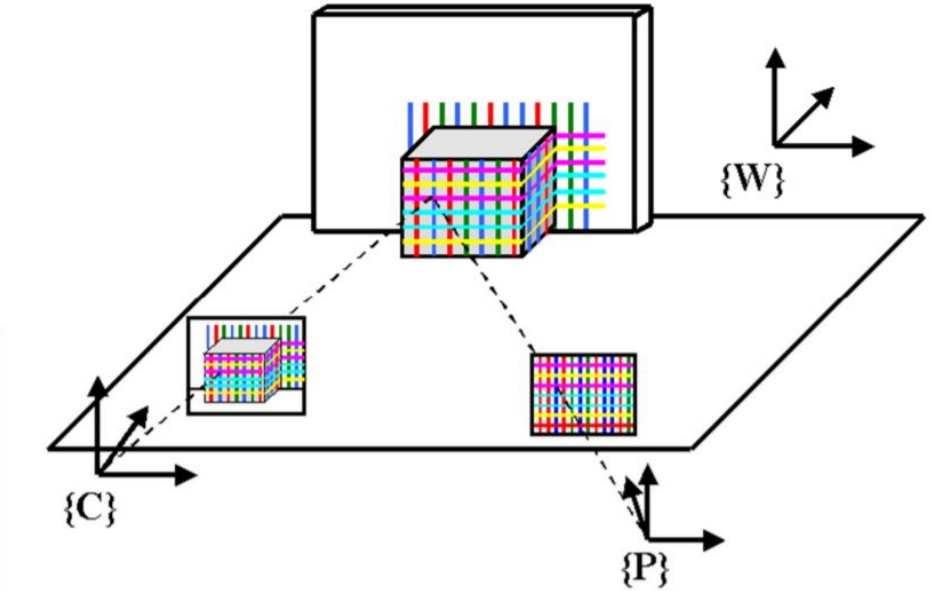
Structured Light 3D Scanning

Sarah Pethani

Structured Light Scanning

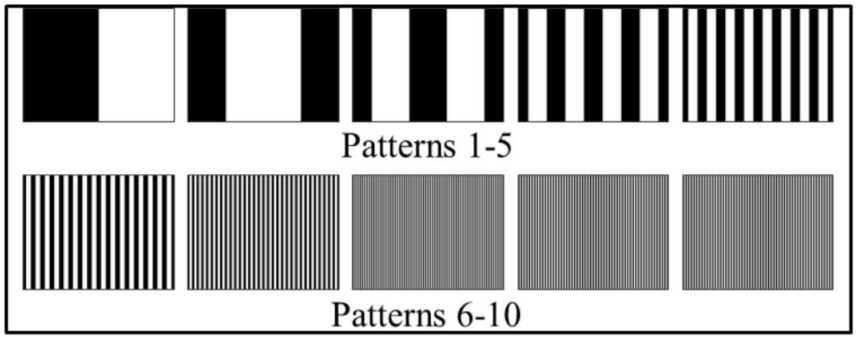
Triangulation:

- Replace a camera with a projector
- Solve correspondence problem by searching the pattern in the camera image



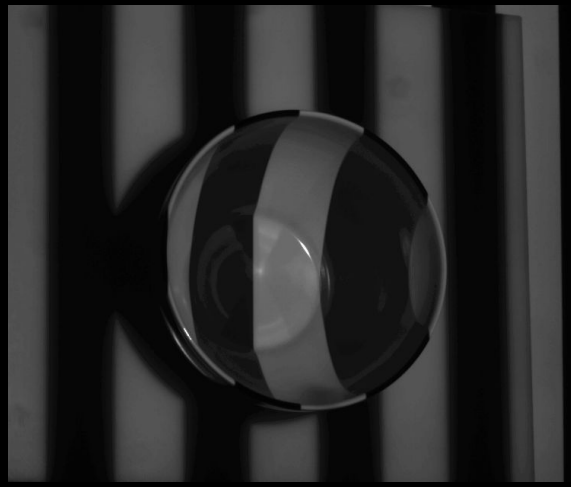
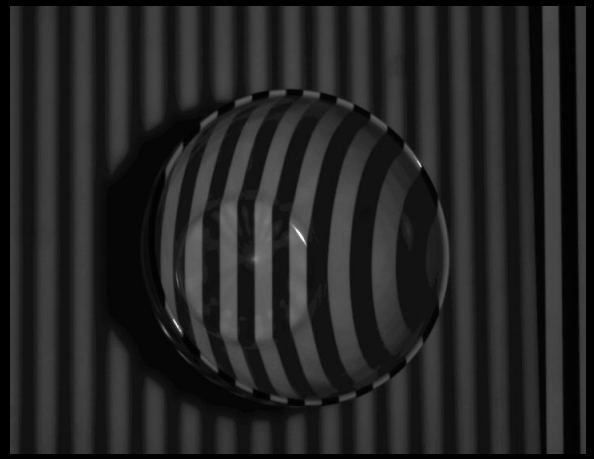
Time Multiplexing: Temporal Binary Codes

- Time Multiplexing: create a codeword by projecting many patterns on scene
  - First projected pattern is most often the most significant bit
  - Follow coarse-to-fine paradigm
  - This process is called “decoding”
- Temporal binary codes: display sequence of white and black bars
  - White bar = 1, black bar = 0



(b) Conventional Gray codes





Problems with Basic Structured Light Scanning

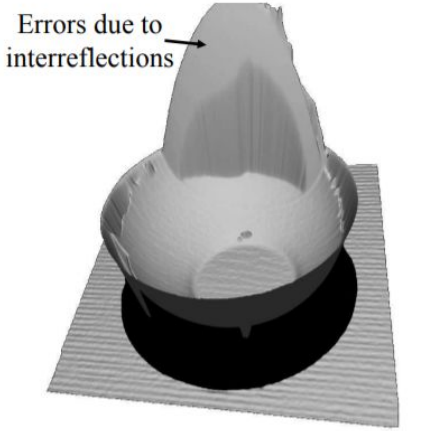
- Incident illumination is large problem for surface reconstruction
  - Short-range effects: subsurface scattering and defocus
  - Long-range effects: interreflections



(a) Styrofoam cup



(b) Conventional Gray codes



(d) Shape recovered using
Conventional Gray codes (11 images)

How to fix?

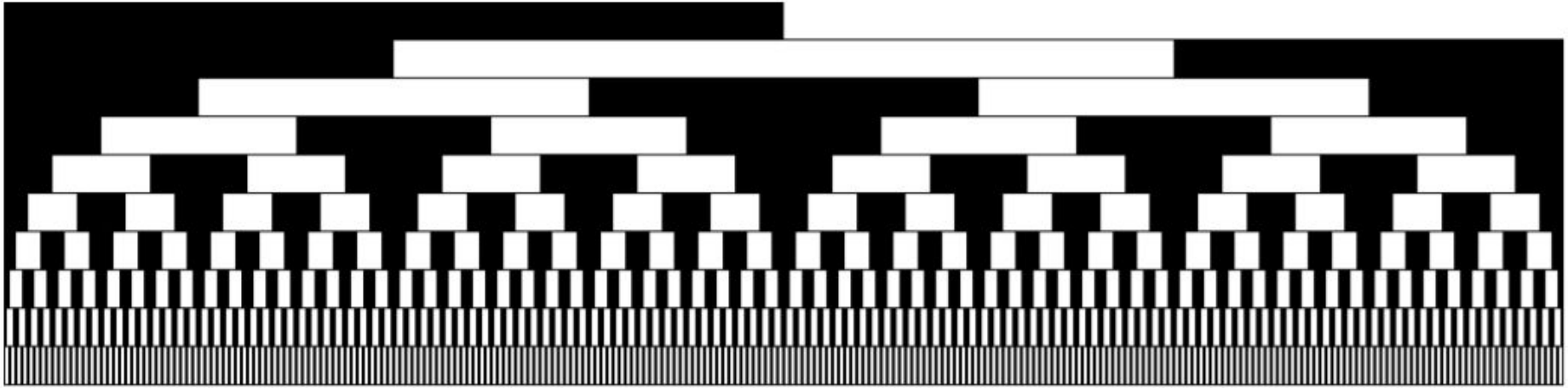
- For short range effects, want to use ONLY low-frequency patterns
- For long range effects, want to use ONLY high-frequency patterns

- Decompose low-frequency pattern into multiple high-frequency patterns, and project these instead
  - When performing xor on the binary codes resulting from these patterns, we result in binary code for low-frequency pattern
  - Allows us to pick between low-frequency and high-frequency in different scenarios, without projecting a different set of patterns

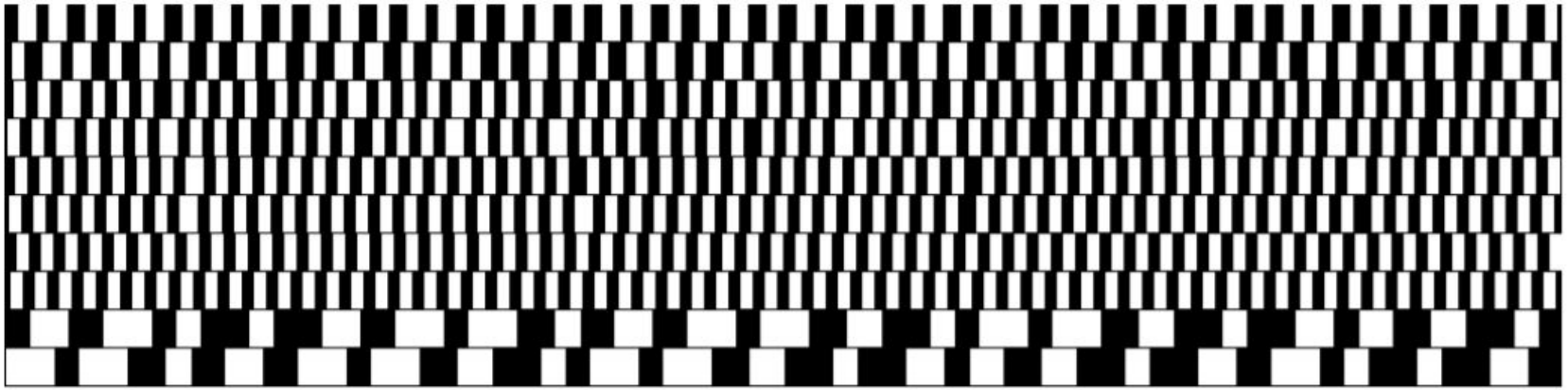
Deciding between low-freq and high-freq

- Project 2 low-freq patterns, and 2 high-freq patterns
- In case of short-range effects:
  - The low-freq patterns will agree with one another and be correct
- In case of long-range effects:
  - The high-freq patterns will agree with one another and be correct

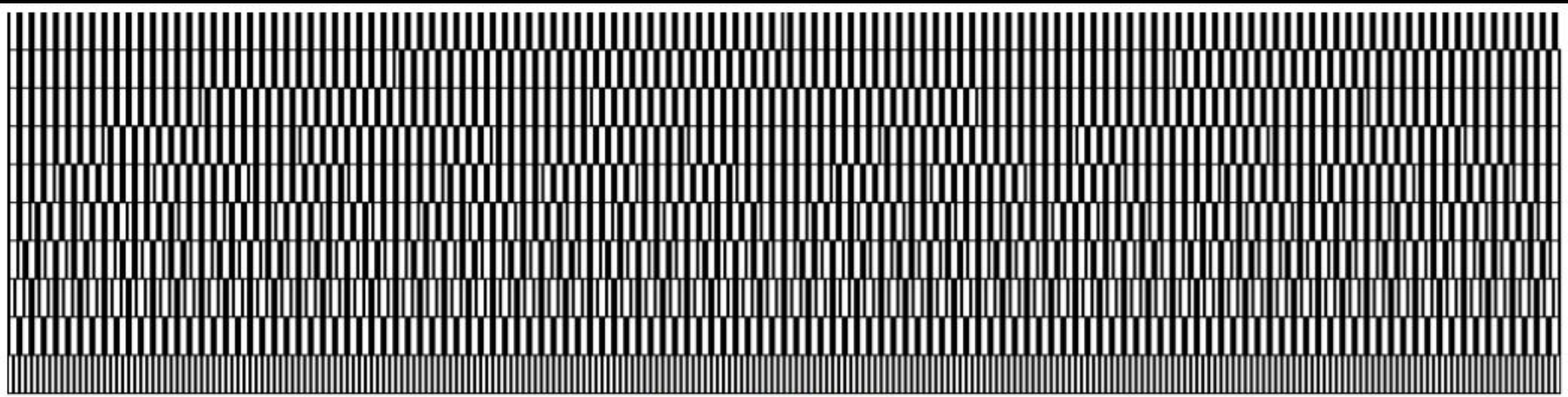
- Intuitively, just perform a consistency check to decide which depth is correct



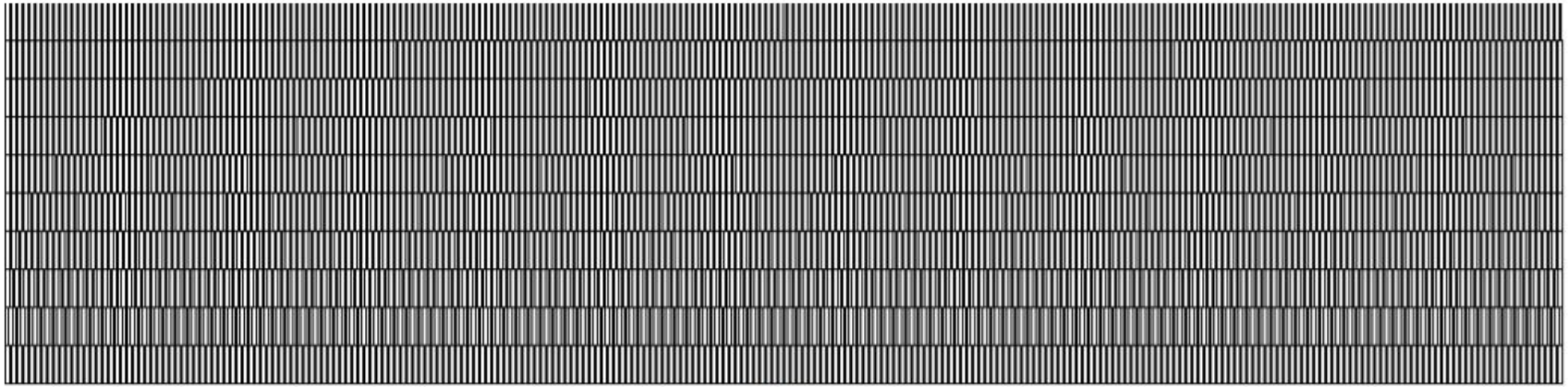
Conventional Gray Codes



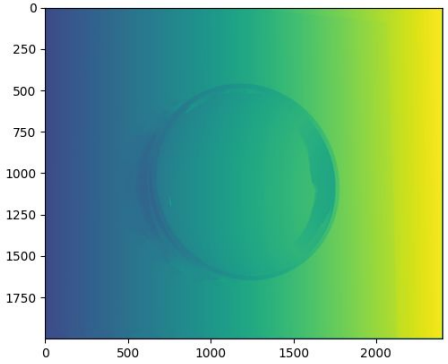
Maximum Min-SW Gray Codes



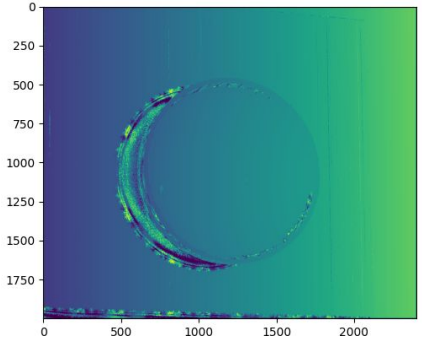
Logical XOR-04 Codes



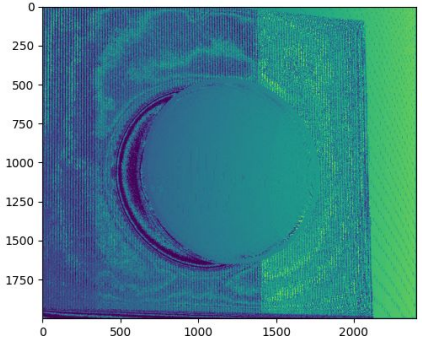
Logical XOR-02 Codes



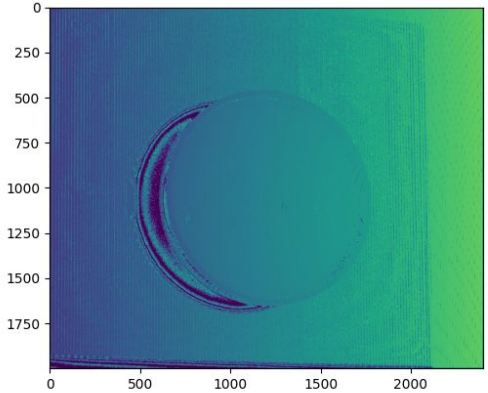
Conventional Gray



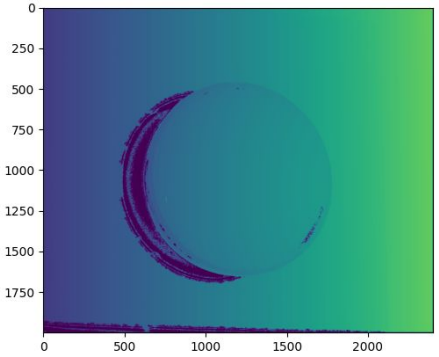
Max-Min SW Gray



XOR 02



XOR 04



Combined

Thanks!

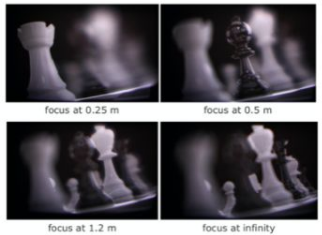
HOLOGRAPHIC PROJECTOR FOR NEAR-EYE DISPLAYS

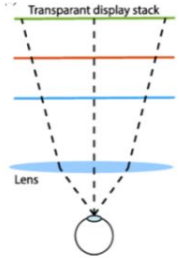
Andrew Maimone, Andreas Georgiou, and Joel S. Kollin. 2017. Holographic near-eye displays for virtual and augmented reality. ACM Trans. Graph. 36, 4, Article 85 (July 2017), 16 pages.

Yingsi Qin

3D DISPLAYS

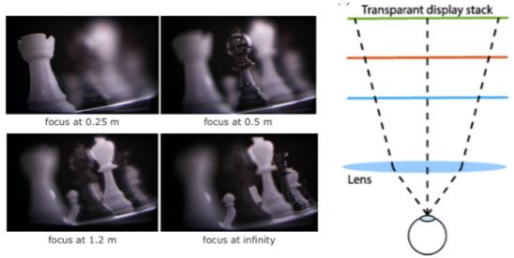
Multifocal Display



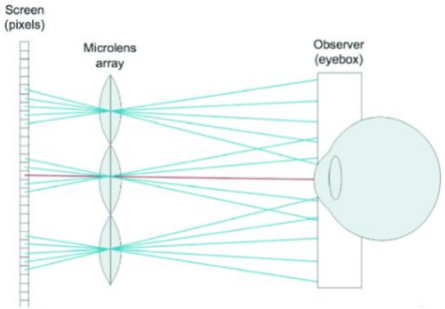


3D DISPLAYS

Multifocal Display

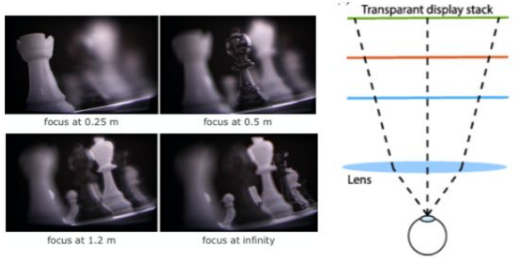


Light Field display

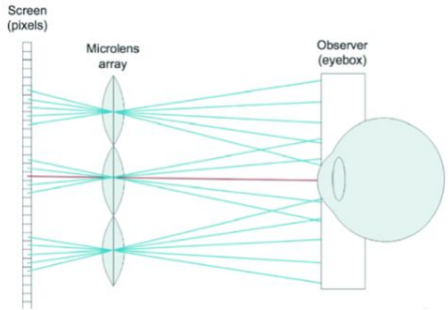


3D DISPLAYS

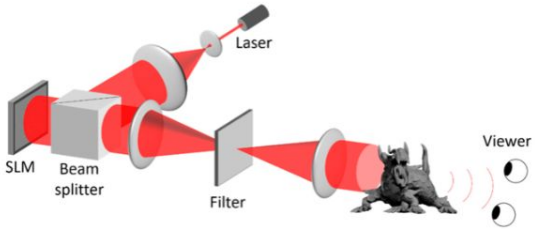
Multifocal Display



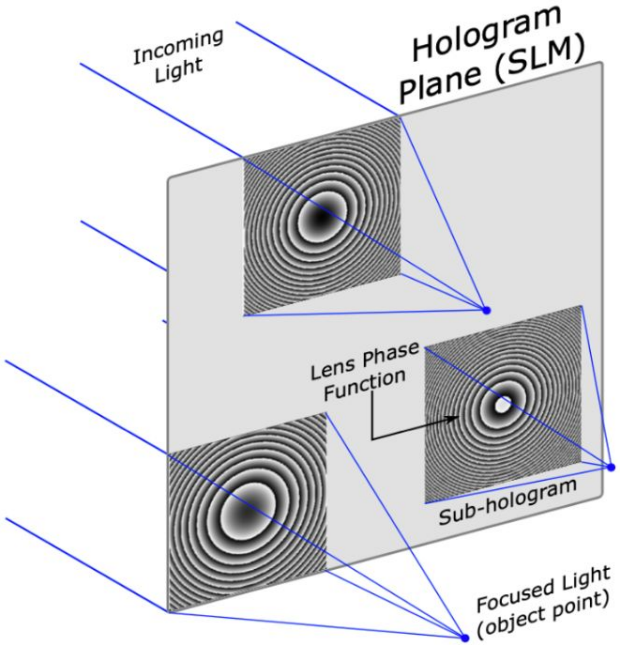
Light Field display



Holographic display



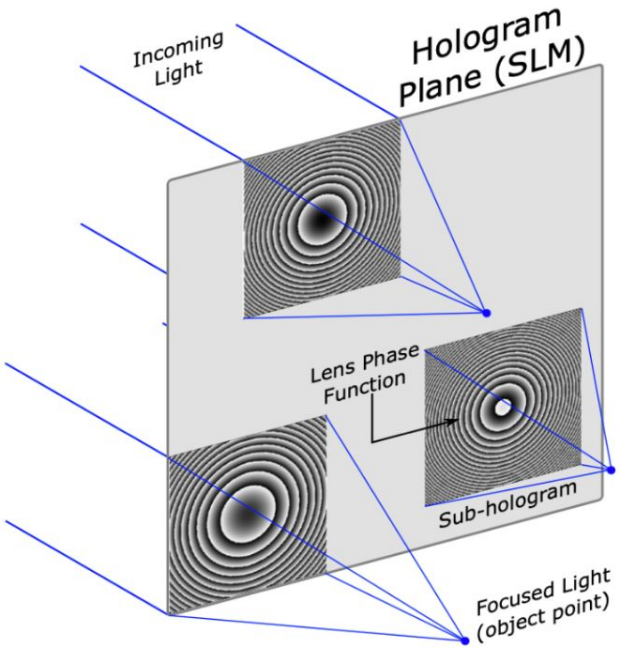
FRESNEL HOLOGRAPHY



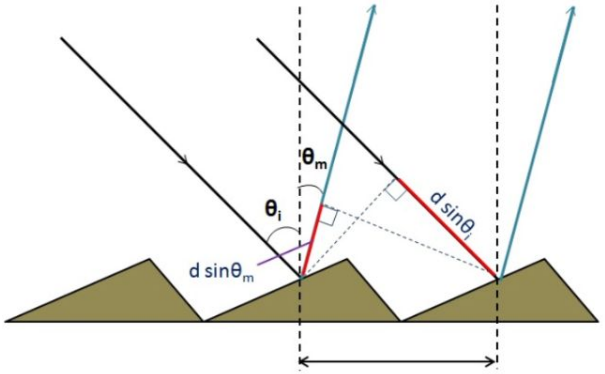
lens phase function:

$$f(\vec{p})_o = e^{i(\phi_o + \frac{2\pi\|\vec{p}-\vec{o}\|}{\lambda})}$$

LIGHT DEFLECTION ANGLE IS RELATED TO SPATIAL FREQUENCY OF A GRATING...



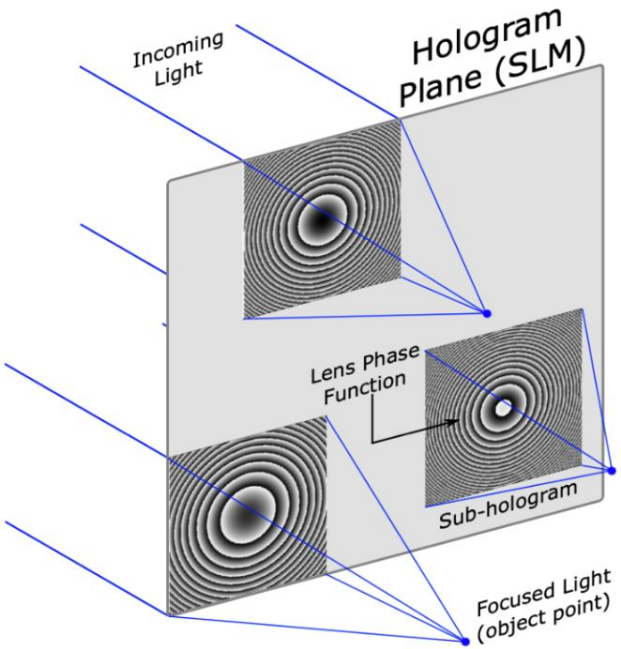
$$d(sin\theta_i + sin\theta_m) = \lambda$$



grating pitch d

our SLM has a 3.74 µm pixel pitch — use sub-hologram!

FRESNEL HOLOGRAPHY



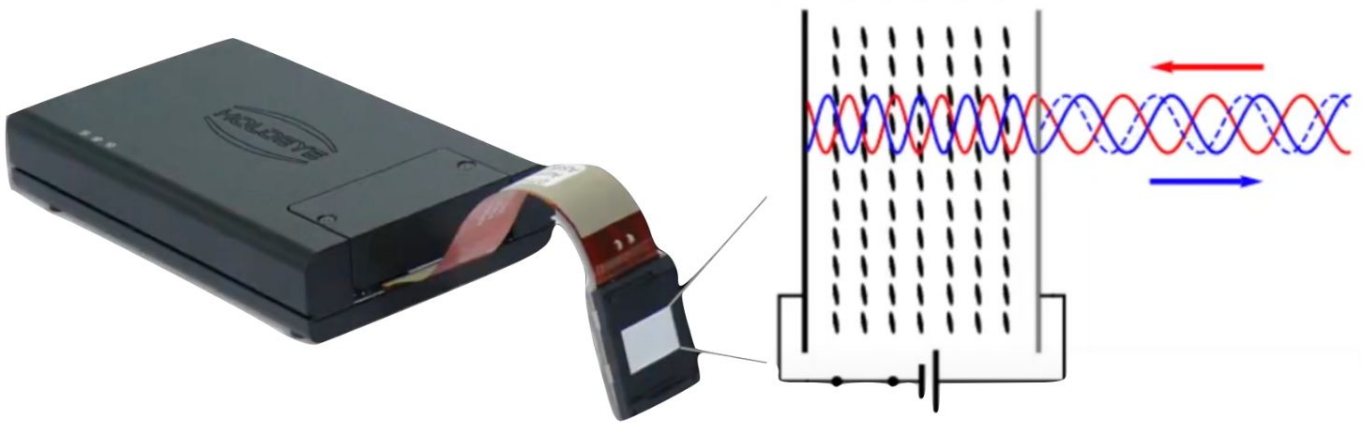
lens phase function:

$$f(\vec{p})_o = e^{i(\phi_o + \frac{2\pi\|\vec{p}-\vec{o}\|}{\lambda})}$$

superposition via point-wise integration:

$$H(\vec{p}) = \sum_{j \in s_p} a_j f(\vec{p})_j$$

BUT HOW TO ENCODE AMPLITUDE WITH A PHASE-ONLY SLM?

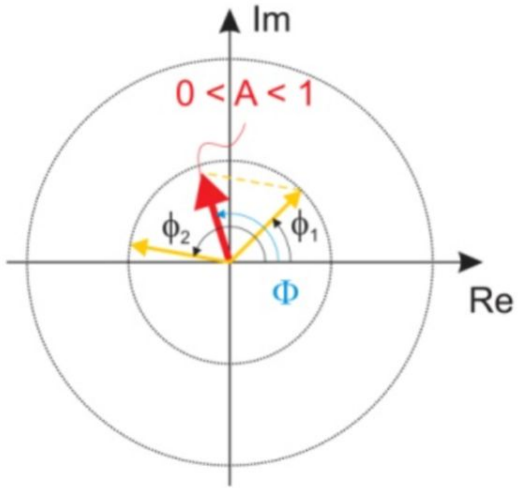


Phase-only modulation

CONVERT TO PHASE-ONLY REPRESENTATION

Double Phase Encoding

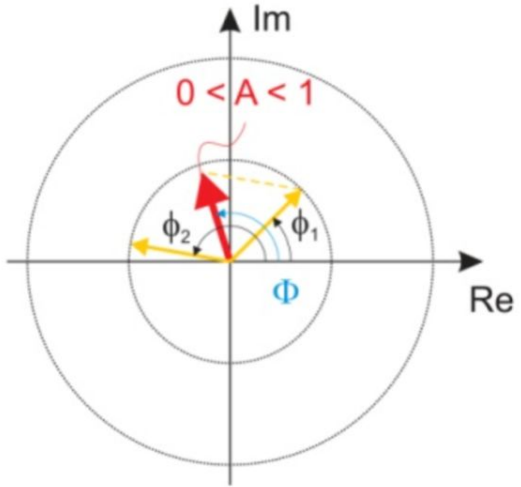
$$Ae^{i\phi} = 0.5e^{i(\phi - \cos^{-1}A)} + 0.5e^{i(\phi + \cos^{-1}A)}$$



CONVERT TO PHASE-ONLY REPRESENTATION

Double Phase Encoding

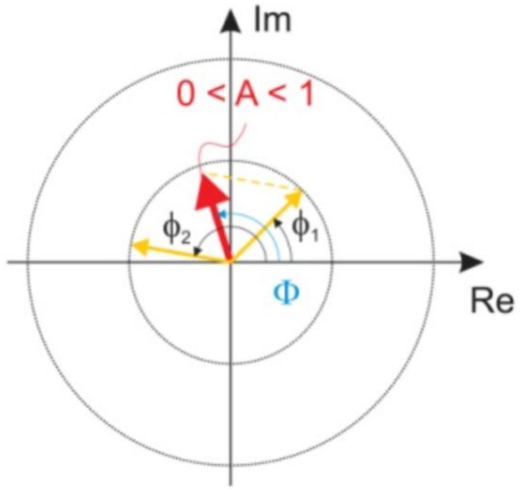
$$Ae^{i\phi} = 0.5e^{i(\phi - \cos^{-1} A)} + 0.5e^{i(\phi + \cos^{-1} A)}$$

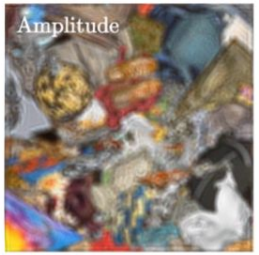


CONVERT TO PHASE-ONLY REPRESENTATION!

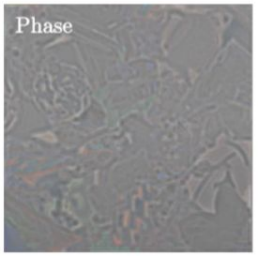
Double Phase Encoding

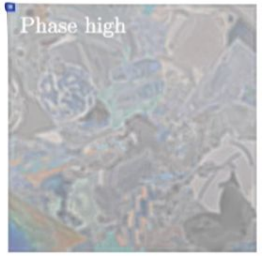
$$Ae^{i\phi} = 0.5e^{i(\phi - \cos^{-1} A)} + 0.5e^{i(\phi + \cos^{-1} A)}$$

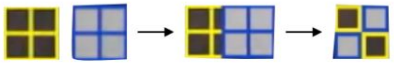










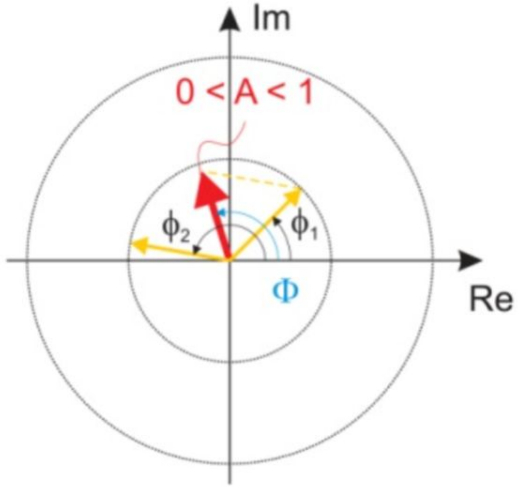


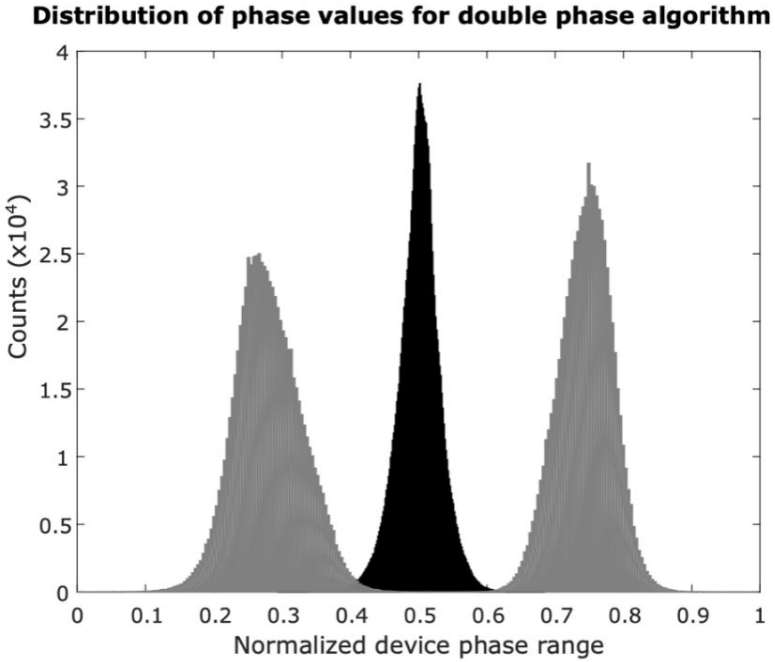
require two phase values for each complex value — expect a corresponding loss in resolution

CONVERT TO PHASE-ONLY REPRESENTATION!

Double Phase Encoding

$$Ae^{i\phi} = 0.5e^{i(\phi - \cos^{-1} A)} + 0.5e^{i(\phi + \cos^{-1} A)}$$

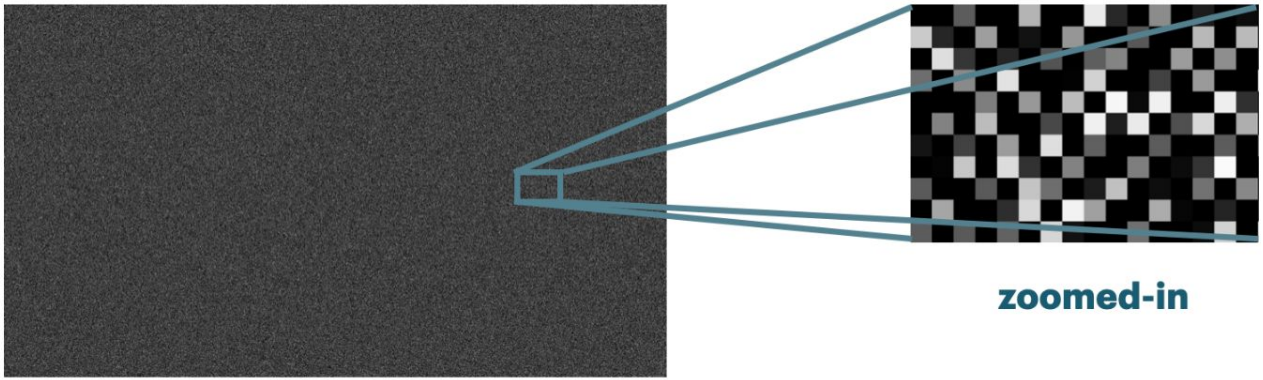




require two phase values for each complex value — expect a corresponding loss in resolution

DOUBLE PHASE ENCODING

$$Ae^{i\phi} = 0.5e^{i(\phi - \cos^{-1}A)} + 0.5e^{i(\phi + \cos^{-1}A)}$$

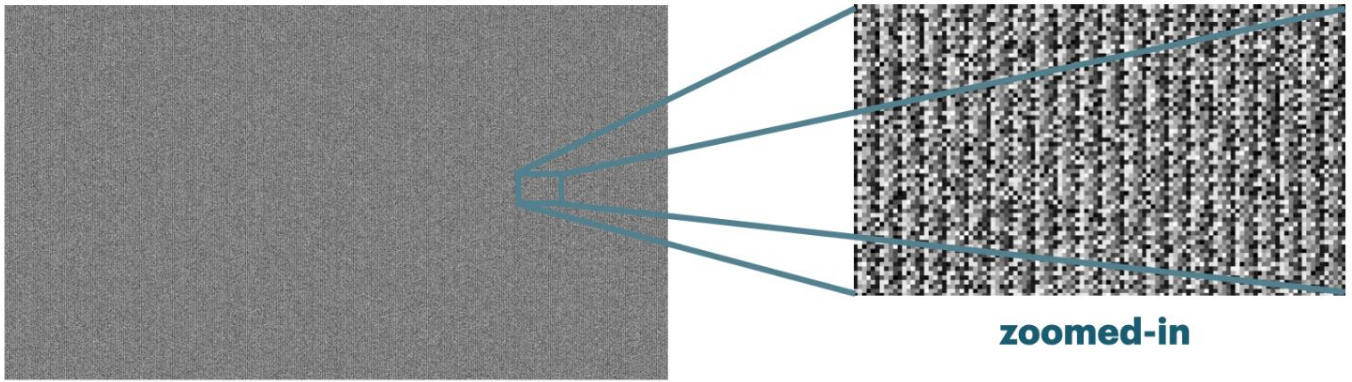


zoomed-in

CGH on SLM

REMOVING THE DC TERM

add a linear phase ramp grating:



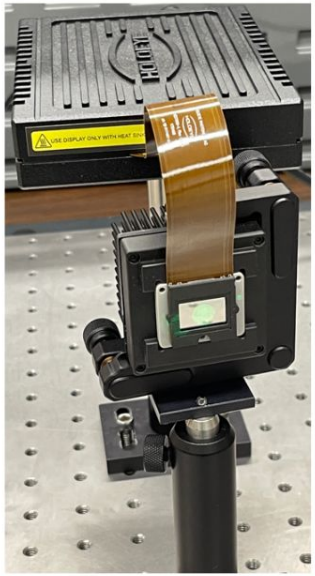
zoomed-in

CGH on SLM

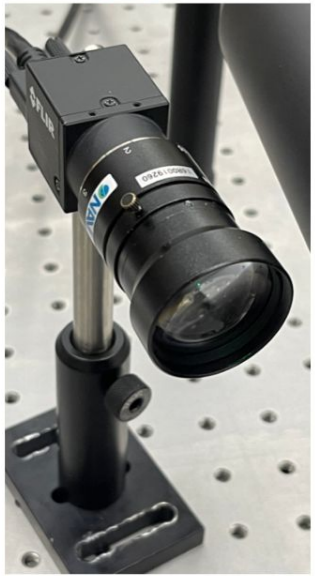
THE SETUP



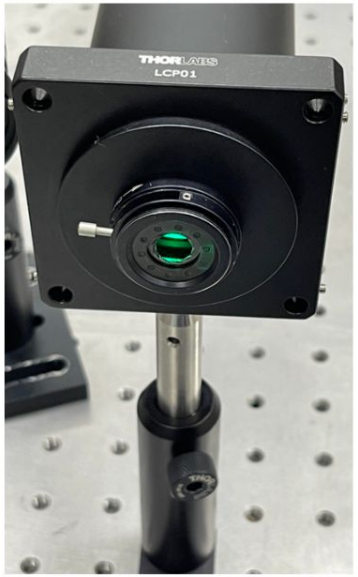
SLM



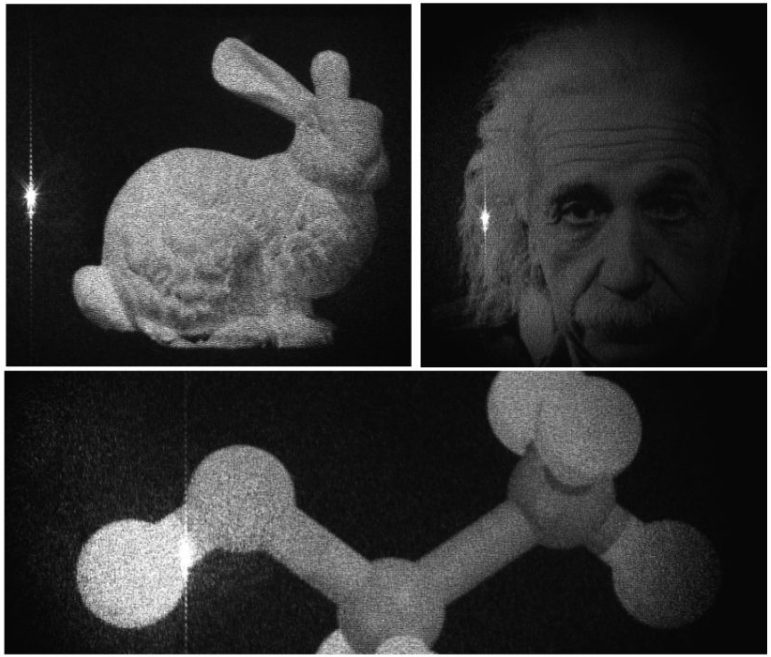
eyepiece

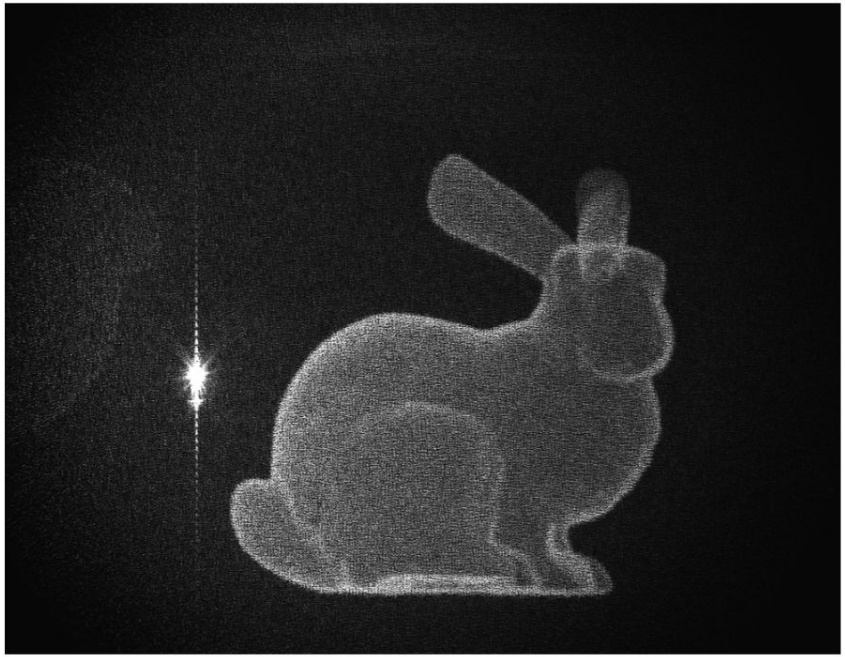


laser aperture



CAPTURED RESULTS





WHAT ARE THE CAUSES OF NOISE?

1. The phase offset term

$$f(\vec{p})_o = e^{i(\phi_o + \frac{2\pi\|\vec{p}-\vec{o}\|}{\lambda})}$$ **lens phase function**

a random constant for 2D

need adjustments for 3D : $$\phi_o = -\frac{2\pi(\|\vec{p_{c_o}} - \vec{l}\| + \|\vec{p_{c_o}} - \vec{o}\|)}{\lambda}$$

2. double-phase encoding

3. pixel-pitch restriction on sub-hologram size

FUTURE WORK

- Adjust phase offset term to improve SNR
- Enable full color display by using 3 lasers (RGB) and an Arduino to switch between them at 20Hz
- Use a diverging light source

THANK YOU

Complete 3D Object Reconstruction from Surround Structured Light Scanning

Nathan Riopelle

Project Goal (As Proposed)

Create a **3D reconstruction** of an object using a single camera under the presence of **structured light**. While traditional approaches for capturing a 360-degree scan of an object require a turntable and multiple images, this project will accomplish the same using a pair of **planar mirrors** and an **orthographic projector** constructed with a Fresnel lens.

Project Goal (As Proposed)

Based off of 2007 3DIMPVT paper

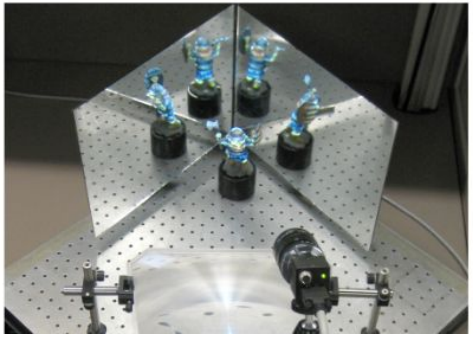
Orthographic projector and planar mirrors generate four “virtual” cameras

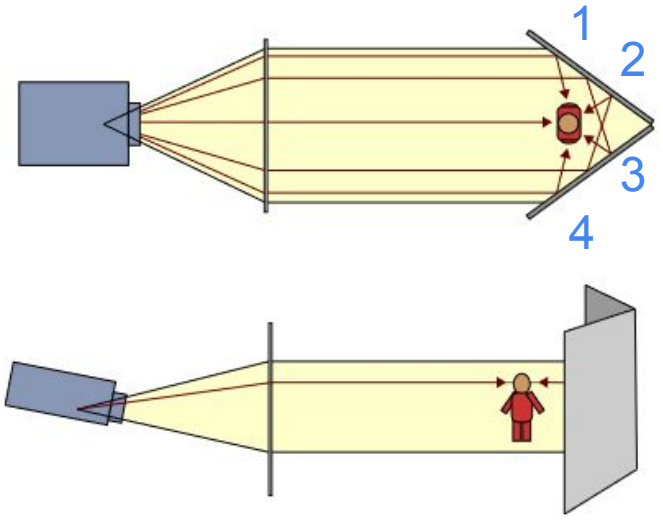
Surround Structured Lighting for Full Object Scanning

Douglas Lanman, Daniel Crispell, and Gabriel Taubin
Department of Engineering, Brown University
{dlanman,daniel_crispell,taubin}@brown.edu

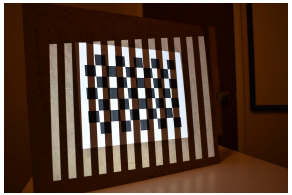
Abstract

This paper presents a new system for acquiring complete 3D surface models using a single structured light projector, a pair of planar mirrors, and one or more synchronized cameras. We project structured light patterns that illuminate the object from all sides (not just the side of the projector) and are able to observe the object from several vantage points simultaneously. This system requires that projected planes of light be parallel, and so we construct an orthographic projector using a Fresnel lens and a commercial DLP projector. A single Gray code sequence is used to encode a set of vertically-spaced light planes within the scanning volume, and five views of the illuminated object





Calibration is Immensely Complex

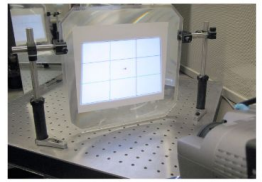


Intrinsic camera calibration

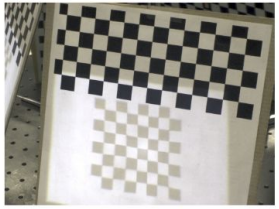
↓

Intrinsic projector calibration

↓



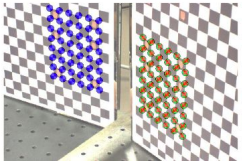
Projector / Fresnel lens focal point alignment



↓

Mapping of scan lines to planes in 3D

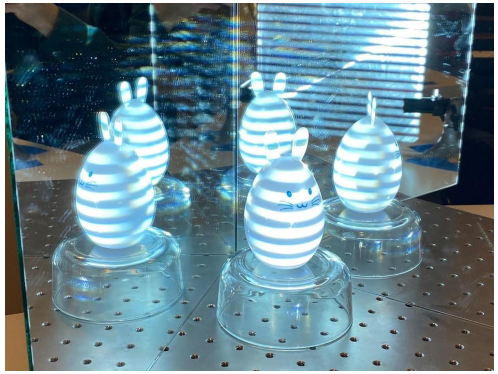
↓



Estimation of position/pose of mirrors relative to camera

First Attempt at Surround Scanning







Revised Project Goal

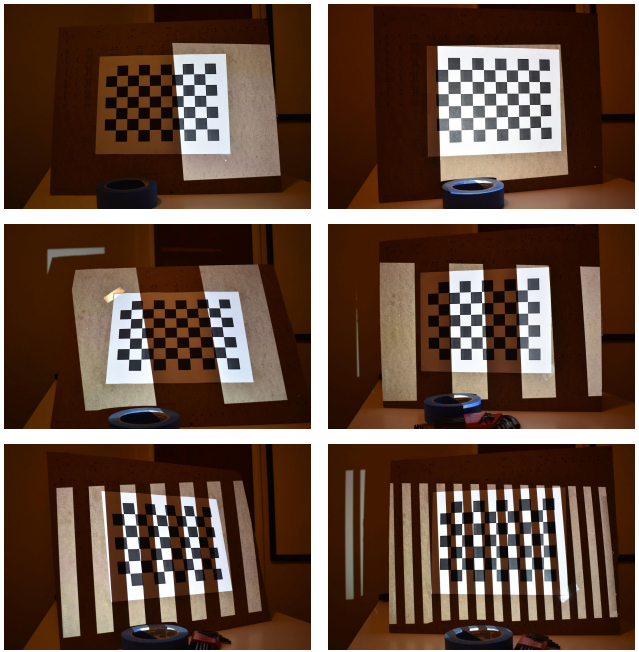
Create a single-view **3D reconstruction** of an object using a single camera under the presence of **structured light**. As a stretch goal, attempt surround scanning using a pair of **planar mirrors** and an **orthographic projector** constructed with a Fresnel lens.

Single-View Scanning Setup

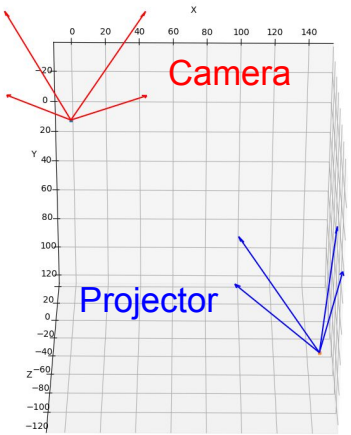




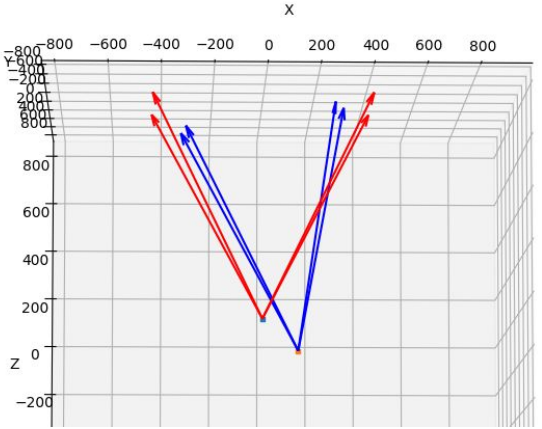
Intrinsic Camera + Projector Calibration



Calibrated by projecting structured light
Gray codes onto a 9x6 checkerboard



Zoomed in



Zoomed out

3D Scan Targets



Larger,
Simple geometry

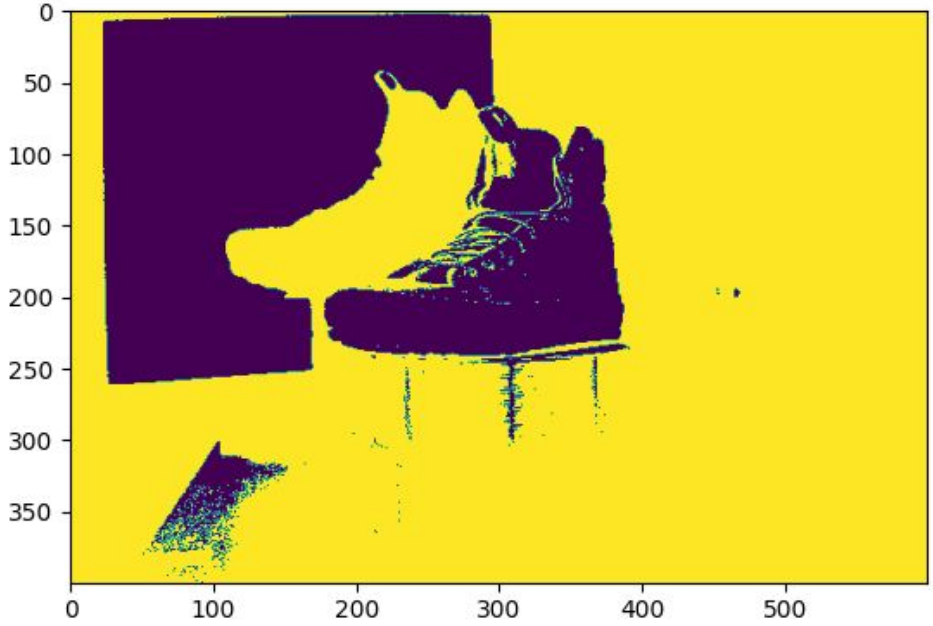


Many inter-occlusions,
Detailed



Large depth variation in
small area

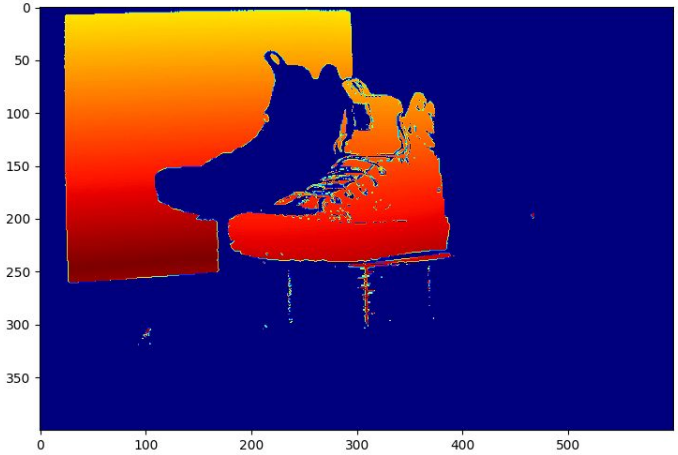
Projected Area Mask

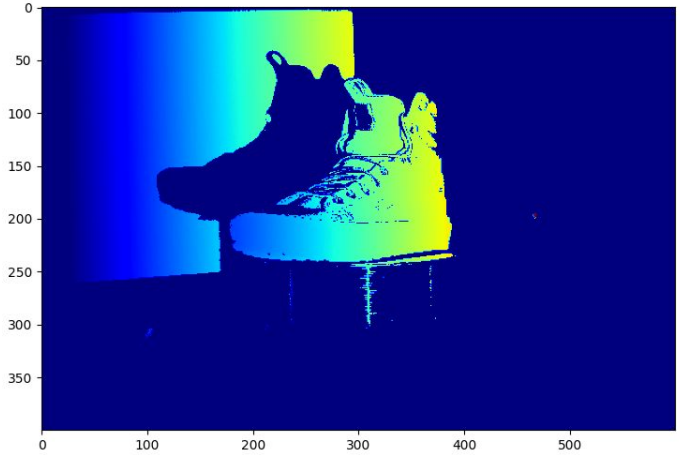


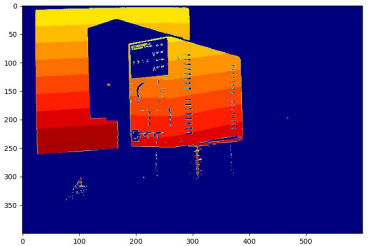
$$if:\ \left(I_{full} - I_{ave}\right) < C_{thres}\ \ then\ \ 0$$

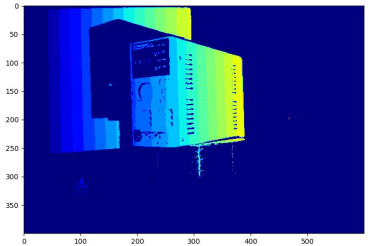
$$else: 1$$

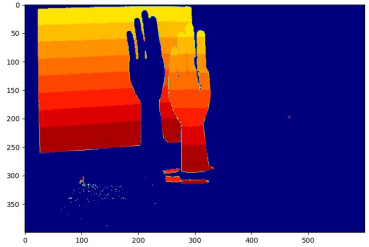
Projector Row + Column Decodings

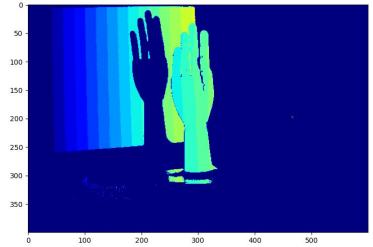




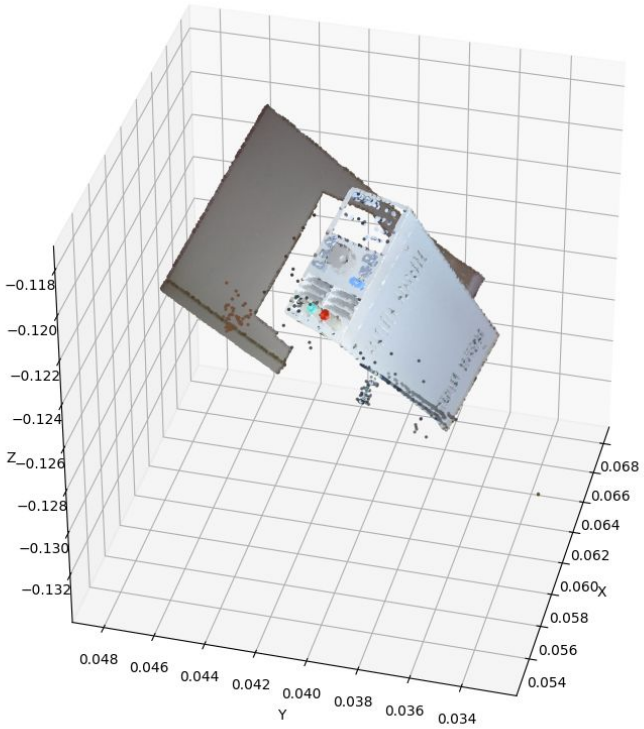


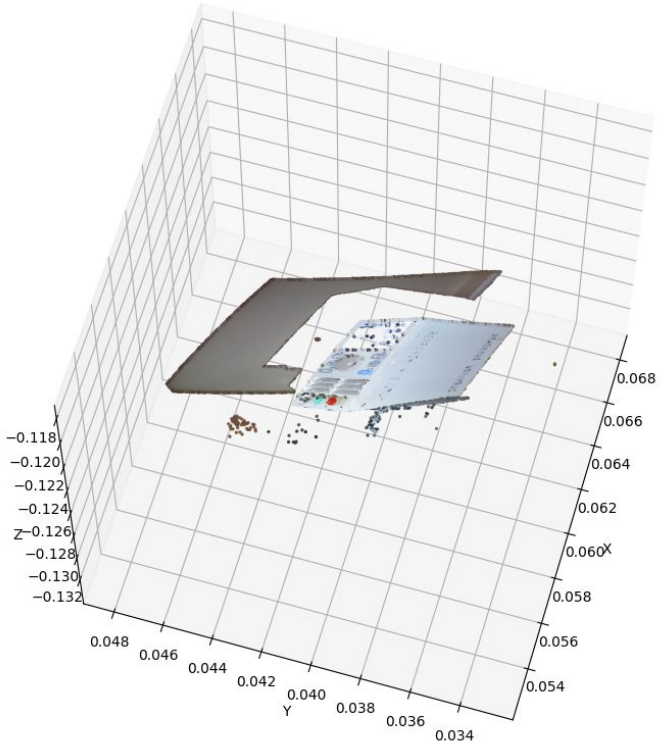




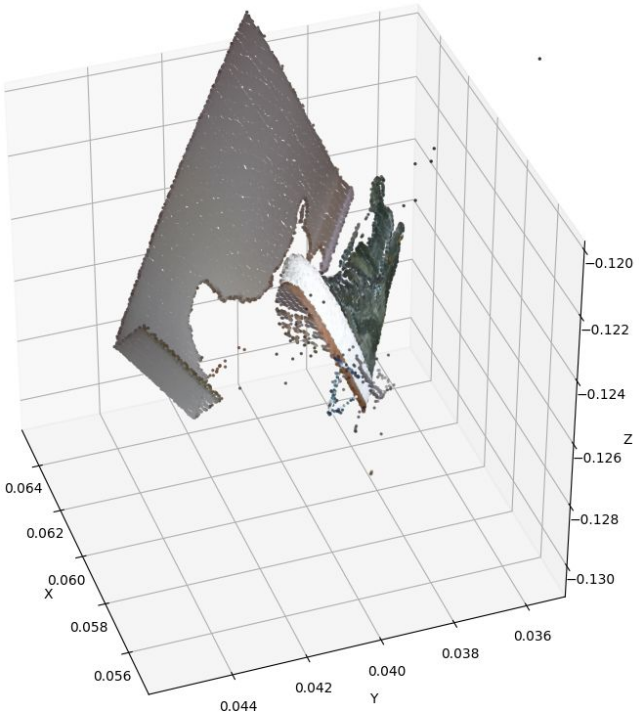


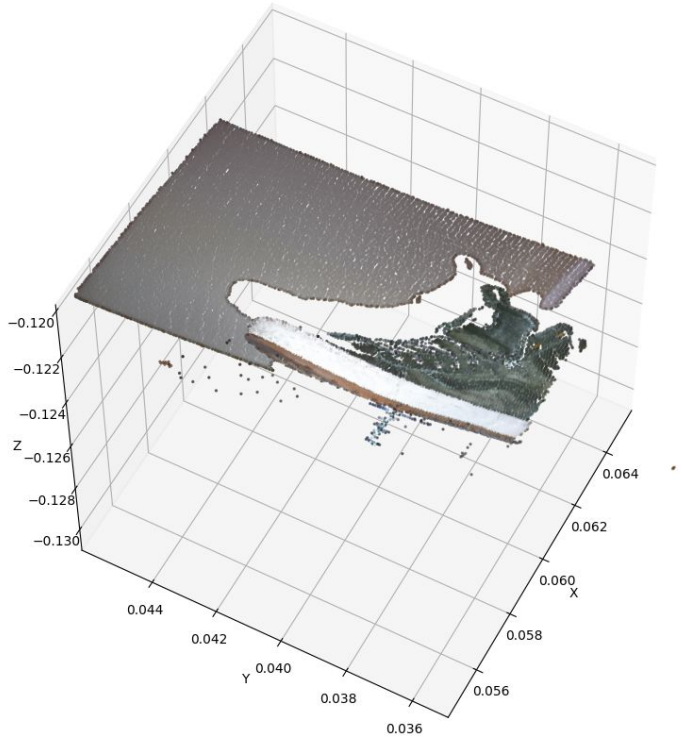
3D Reconstruction Results



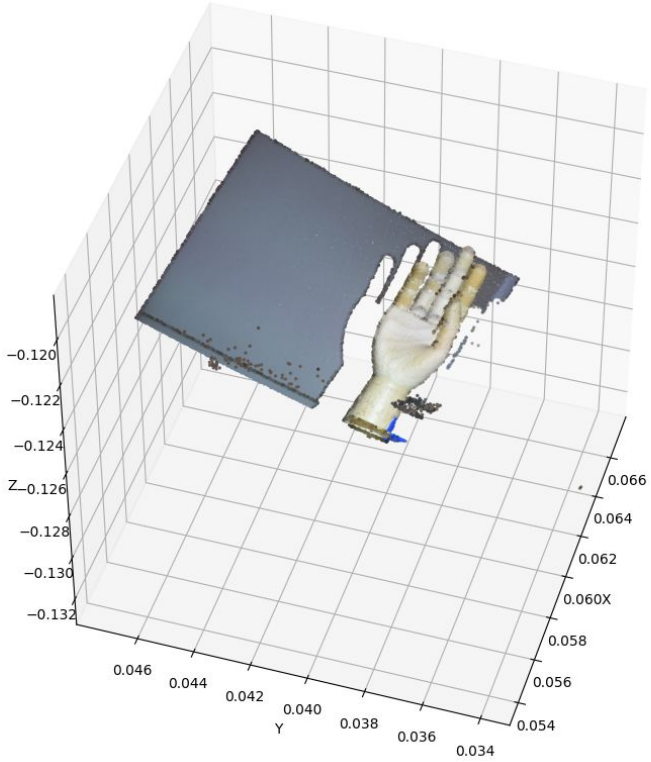


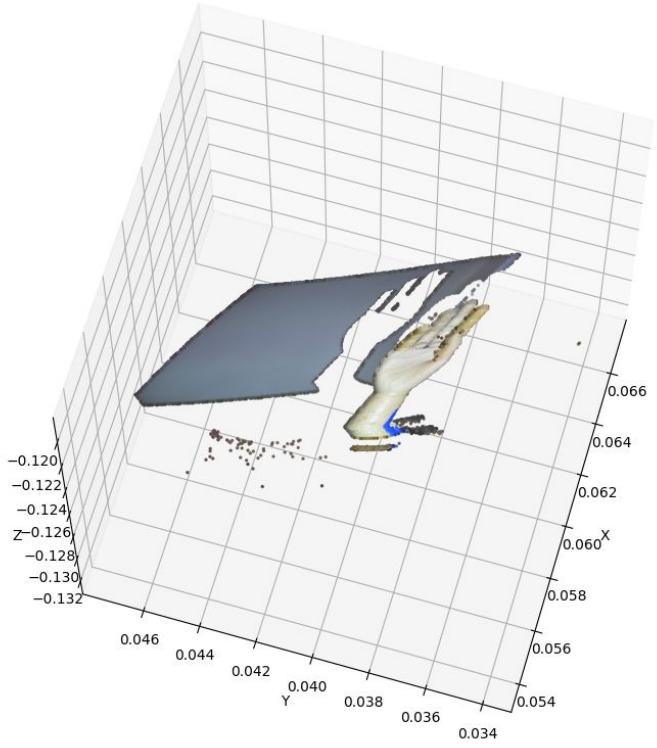
3D Reconstruction Results





3D Reconstruction Results





References

1] Gupta, M., Agrawal, A., Veeraraghavan, A., & Narasimhan, S. G. (2013). A Practical Approach to 3D Scanning in the Presence of Interreflections, Subsurface Scattering and Defocus. *International Journal of Computer Vision*, *102*(1–3), 33–55. https://doi.org/10.1007/s11263-012-0554-3

[2] Gupta, M., Agrawal, A., Veeraraghavan, A., & Narasimhan, S. G. (2011). Structured light 3D scanning in the presence of global illumination. *CVPR 2011*, 713–720. https://doi.org/10.1109/CVPR.2011.5995321

[3] Lanman, D., Crispell, D., & Taubin, G. (2007). Surround Structured Lighting for Full Object Scanning. *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007)*, 107–116. https://doi.org/10.1109/3DIM.2007.57

[4] Lanman, D., & Taubin, G. (2009). Build your own 3D scanner: 3D photography for beginners. *ACM SIGGRAPH 2009 Courses on - SIGGRAPH '09*, 1–94. https://doi.org/10.1145/1667239.1667247

[5] Moreno, D., & Taubin, G. (2012). Simple, Accurate, and Robust Projector-Camera Calibration. *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, 464–471. https://doi.org/10.1109/3DIMPVT.2012.77

[6] *Patent Images*. (n.d.-a). Retrieved October 25, 2021, from https://pdfaiw.uspto.gov/.aiw?PageNum=0&docid=20170206660&IDKey=&HomeUrl=%2F

[7] Salvi, J., Pagès, J., & Batlle, J. (2004). Pattern codification strategies in structured light systems. *Pattern Recognition*, *37*(4), 827–84**9.** **https://doi.org/10.1016/j.patcog.2003.10.002**

IMU-Aided Deblurring with Smartphone

Jason Xu, Sanjay Salem

Premise



Choices to Increase Exposure

- Increase ISO

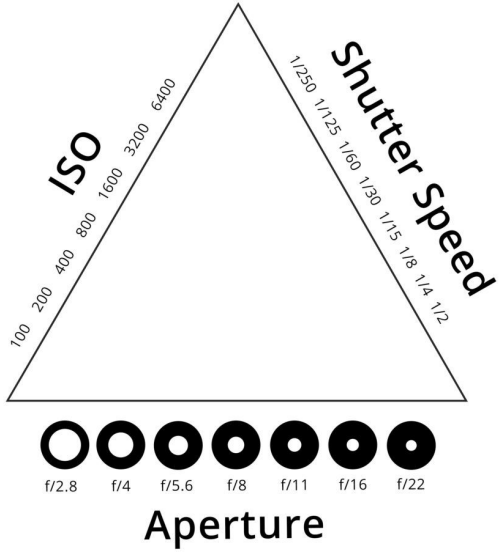
  High amount of noise

- Decrease aperture

  Impossible on smartphones and smaller camera

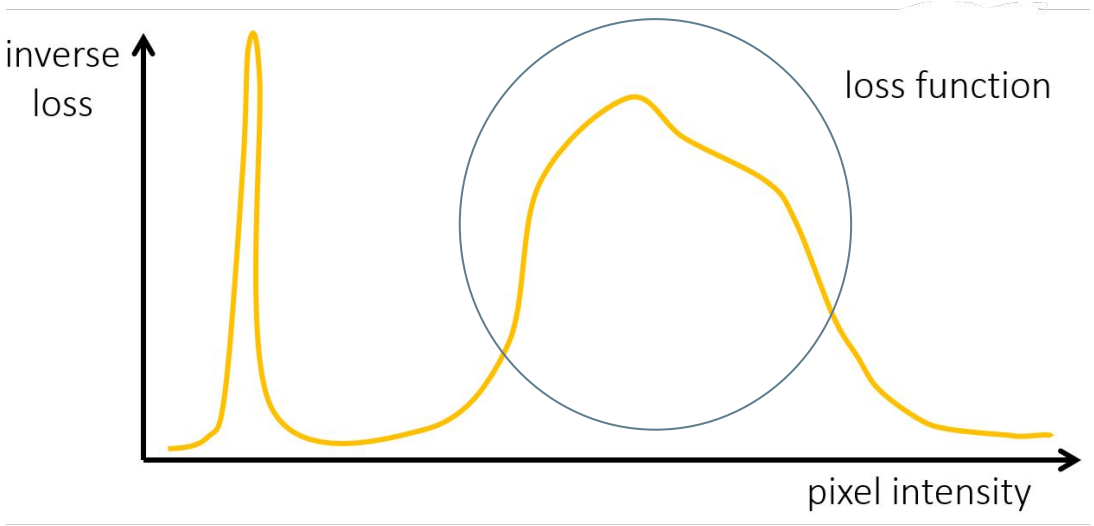
- Decrease shutter speed

  Increased blur (unless you have very steady hands)



Naïve Method - Blind Deconvolution

- Guess blur kernel using various heuristics
- Deconvolve image using regularized least squares optimization
- Lots of assumptions and wide range of plausible solutions



What If...?

We could do better?

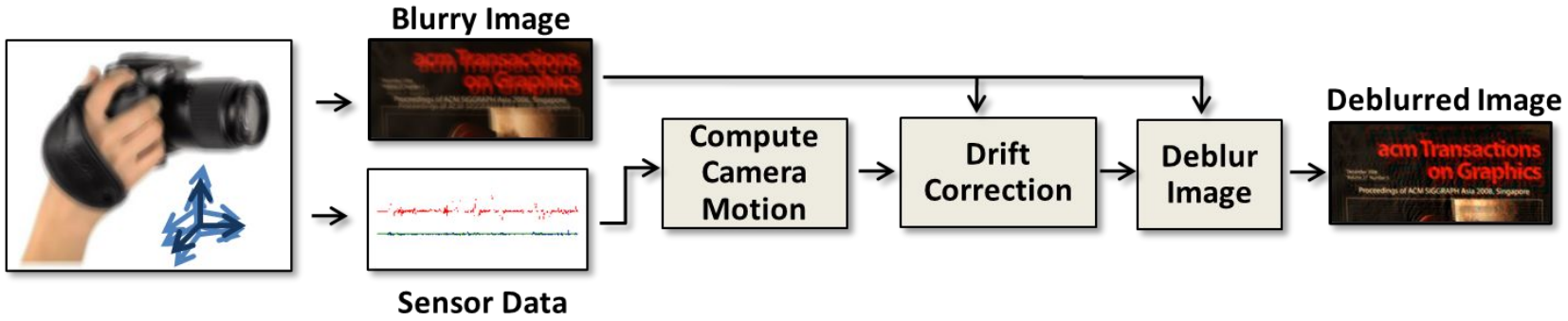
Improved method:
IMU Aided Deconvolution



Joshi, Neel et al. *Image Deblurring using Inertial Measurement Sensors*.

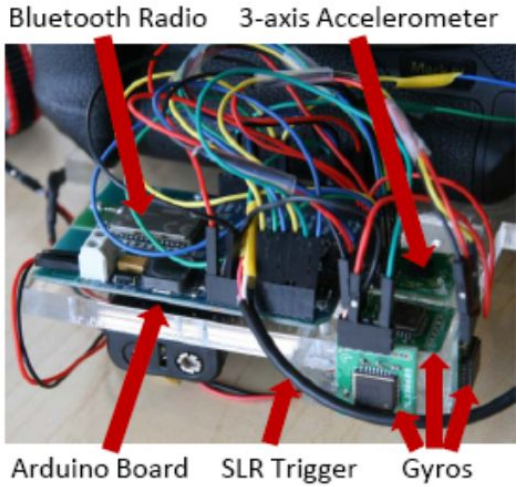
High Level Approach

- Use IMUs to record camera movement
- Compute blur kernel from rotation and translation data
- Use blur kernel from motion to deconvolve image



Our Improvements

- Using smartphone instead of complex capturing rig
- Incorporating magnetometer data to increase DoF





Data Collection

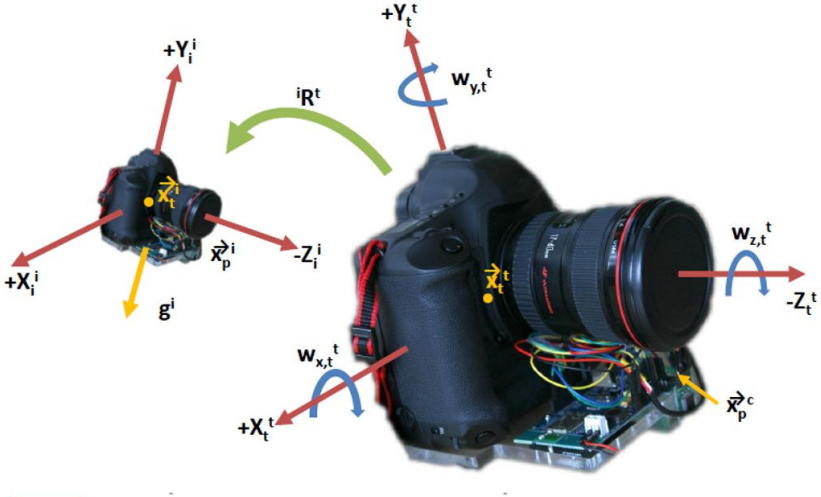
- Android app logging gyroscope, accelerometer and magnetometer data, synchronized to start of exposure
- Exposure time of 240ms



Huai, Jianzhu et al. *Mobile AR Sensor Logger*.

Calculation of Camera Motion

- Integrate linear acceleration twice to get translation data
- Use Madgwick filter with gyro, accelerometer and magnetometer data to get attitude (rotation data)
  - Better method than original paper



$$\vec{\omega}_t^t = {}^tR^i * \vec{\omega}_t^i$$

$$\vec{a}_p^t = {}^tR^i \left(\vec{a}_t^i + \vec{g}^i + (\vec{\omega}_t^i \times (\vec{\omega}_t^i \times \vec{r}_p^q)) + (\vec{\alpha}_t^i \times \vec{r}_p^q) \right)$$

Forming the Blur Kernel

$$A_t \propto H_t = K\left(R_t + \frac{1}{d} T_t N^T\right) K^{-1}$$

Homography — Rotation — Translation — Intrinsics

Planar Depth

Blur/Motion Warping Matrix

$$\vec{I}_t = A_t \vec{I}$$

$$\vec{B} = \sum_{t=0}^{s} A_t \vec{I}$$

Deconvolution

$$I = \arg\min_{I} \left\| B - \sum A_t I \right\|^2 \Big/ \sigma^2 + \lambda \left| \nabla I \right|^{0.8}$$

Regularized Least-Squares Optimization

Example Output





Example Output

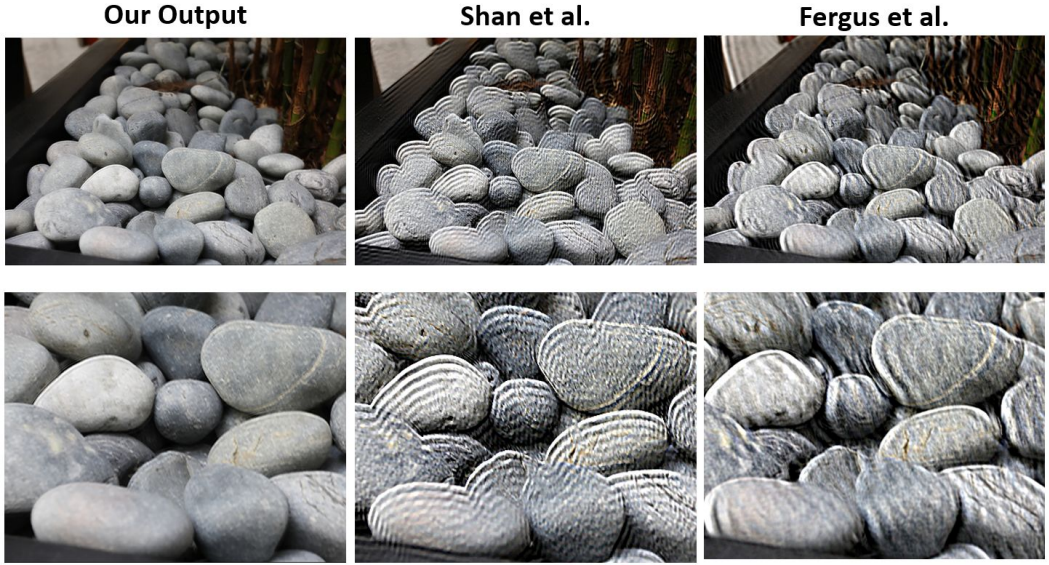


Example Output



Advantages

- Takes the guesswork out of forming the blur kernel
- Less computationally intensive - fewer unknowns
- Better quality



References

- Joshi, Neel, et al. “Image Deblurring Using Inertial Measurement Sensors.” ACM Transactions on Graphics, vol. 29, no. 4, ACM, 2010, pp. 1–9, https://doi.org/10.1145/1778765.1778767.
  - Most formulas and images come from this paper as well.

- Huai, Jianzhu, et al. The Mobile AR Sensor Logger for Android and iOS Devices. 2019. https://github.com/OSUPCVLab/mobile-ar-sensor-logger.
  - Logging app on Android device

Thank you!

Structured Light Blocks-World

Keshav Sangam

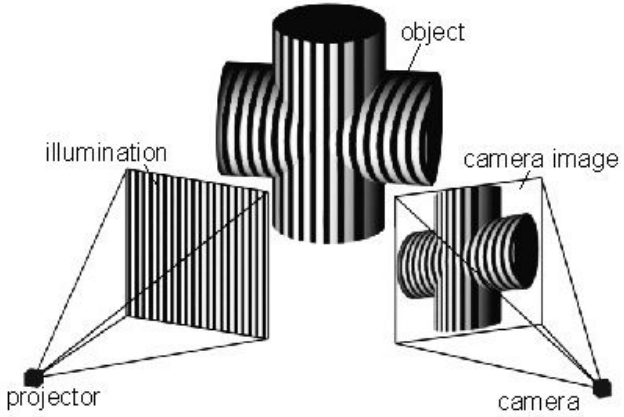
Traditional Structured Light Scanning

Triangulation:

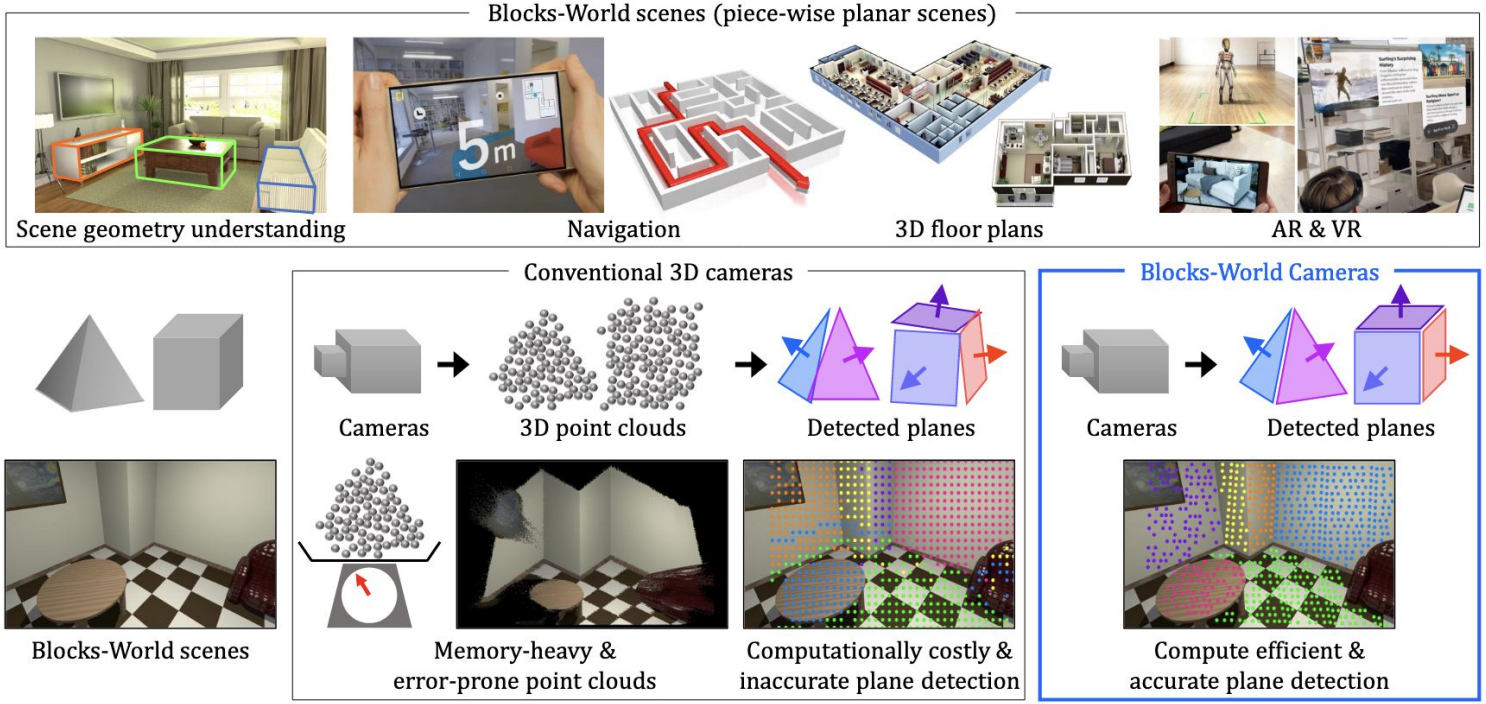
- Replace a camera with a projector
- Solve correspondence problem by searching the pattern in the camera image

Why is this inefficient?

- Correspondence problems traditionally take lots of compute power
- Going from a 3D point cloud to a 2D planar representation wastes memory, and the planes found are often inaccurate
- More accuracy often necessitates multiple images being captured

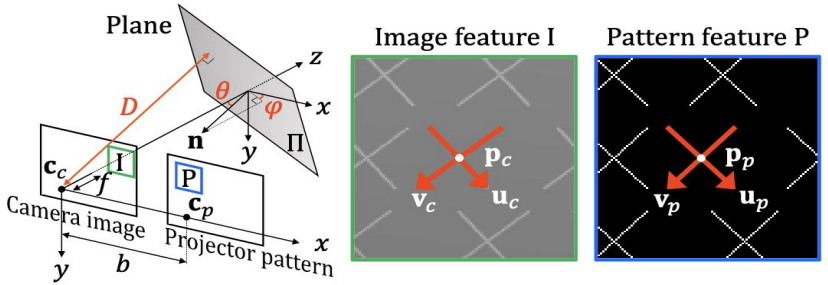


Blocks-World Structured Light Scanning



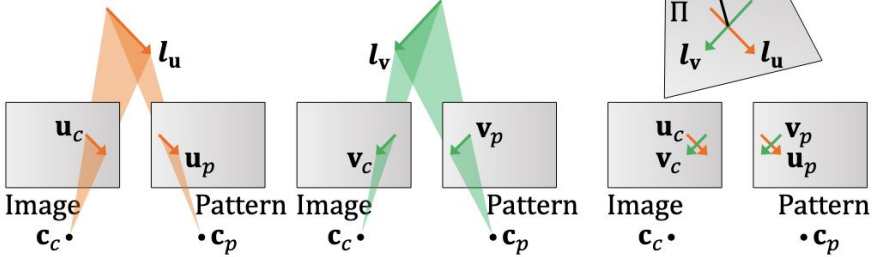
Planes from Known Correspondences

Given a known image feature corresponding to a known pattern feature, we can use geometry to find the plane parameters such as the surface normal and shortest distance D from from the camera origin to the plane.



(a) Imaging system

(b) A pair of pattern and image features



(a) l_u and l_v formed by pairs of features (b) Plane defined by l_u and l_v

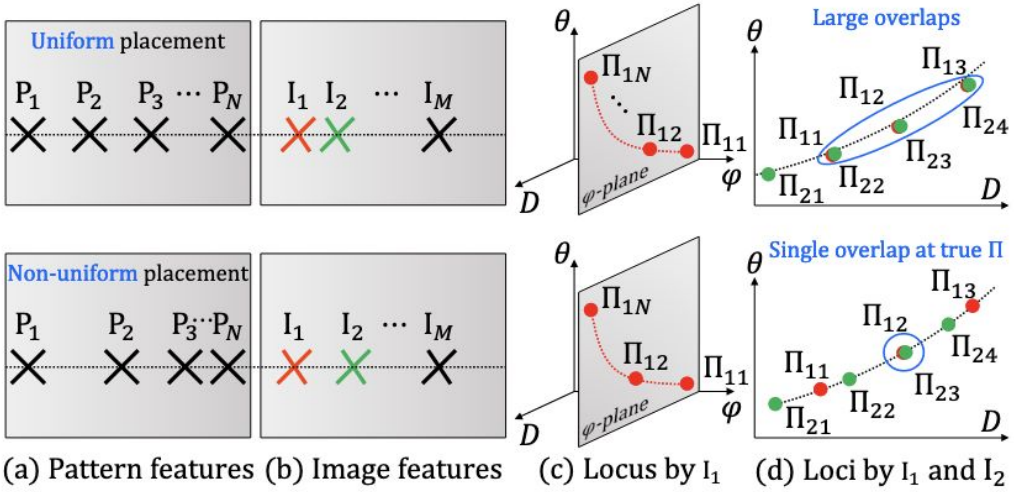
Planes from Unknown Correspondences

5.1. Geometric Approach to Correspondence-Free Plane Estimation

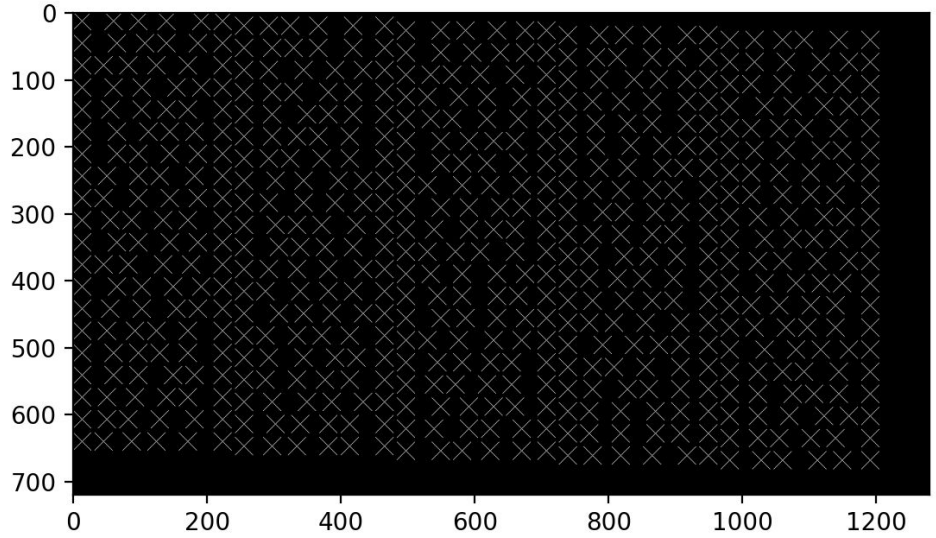
In order to address this tradeoff, we develop a novel, light-weight computational approach for estimating plane parameters *without* explicitly computing correspondences between image and pattern features. Let the set of pattern features on one epipolar line of the projected pattern be $\{\mathrm{P}_1, \ldots, \mathrm{P}_N\}$. A subset of these features are mapped to the camera image, resulting in the set of image features $\{\mathrm{I}_1, \ldots, \mathrm{I}_M\}$ $(M \leq N)$ (upper row of Figs. 4 (a) and (b)).

Consider one image feature, say I_1. All the N pattern features are candidate matching features. Each candidate pattern feature results in a plane hypothesis $\Pi = \{D, \theta, \varphi\}$ by triangulating with the image feature I_1. Accordingly, the set of all candidate pattern features $\{\mathrm{P}_1, \ldots, \mathrm{P}_N\}$ create a set of plane hypotheses $\Lambda_1 = \{\Pi_{11}, \ldots, \Pi_{1N}\}$, where Π_{1n} $(n \in \{1, \ldots, N\})$ is the plane parameters computed from I_1 and P_n. Each plane hypothesis can be represented as a point in the 3D *plane parameter space* (we call this the Π-space), as shown in the upper row of Fig. 4 (c). Therefore, the set of plane hypotheses $\Lambda_1 = \{\Pi_{11}, \ldots, \Pi_{1N}\}$ create a *plane parameter locus* in the Π-space. Similarly, we can create another plane parameter locus $\Lambda_2 = \{\Pi_{21}, \ldots, \Pi_{2N}\}$ by pairing I_2 and $\{\mathrm{P}_1, \ldots, \mathrm{P}_N\}$.

Observation 1. The *key observation* is if I_1 and I_2 correspond to scene points on the same scene plane, then two loci Λ_1 and Λ_2 must intersect. If they intersect at a unique location $\hat{\Pi}$ in the Π-space, then $\hat{\Pi}$ is the true plane parameters.



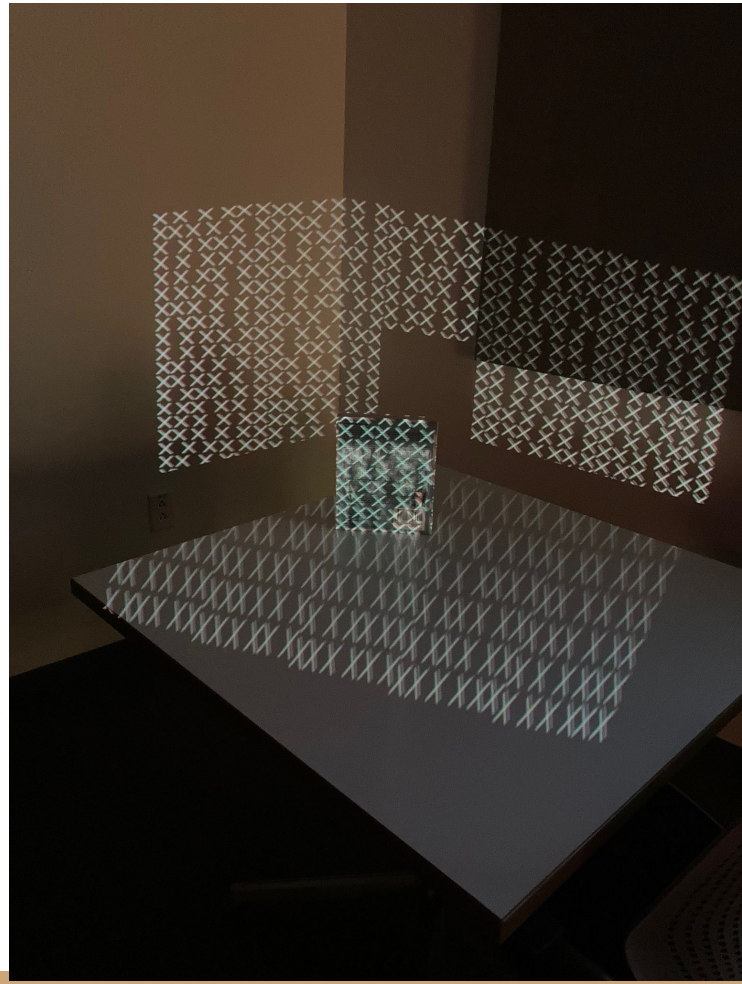
Pattern Design



Did it work?

Not yet; many different problems.





Multi-flash Photobooth

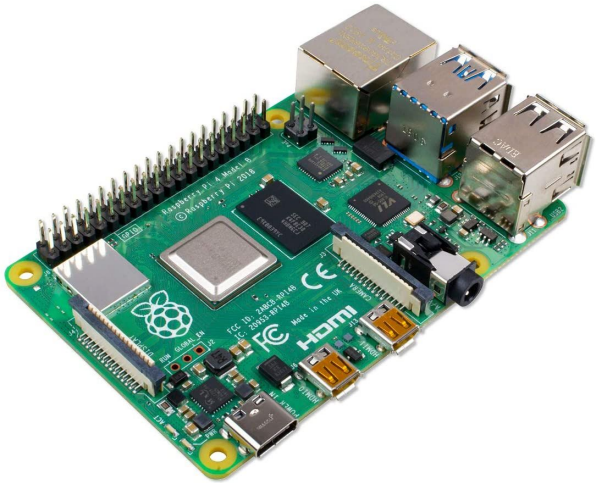
Flash based computational imaging filters

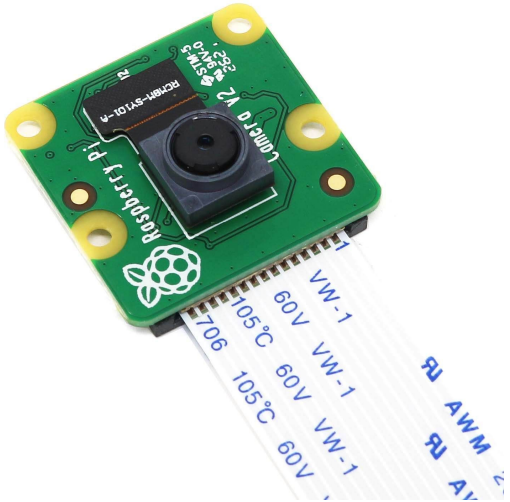
Vivian Shen, vhshen

What can we do by controlling capture conditions with flash?



Hardware setup







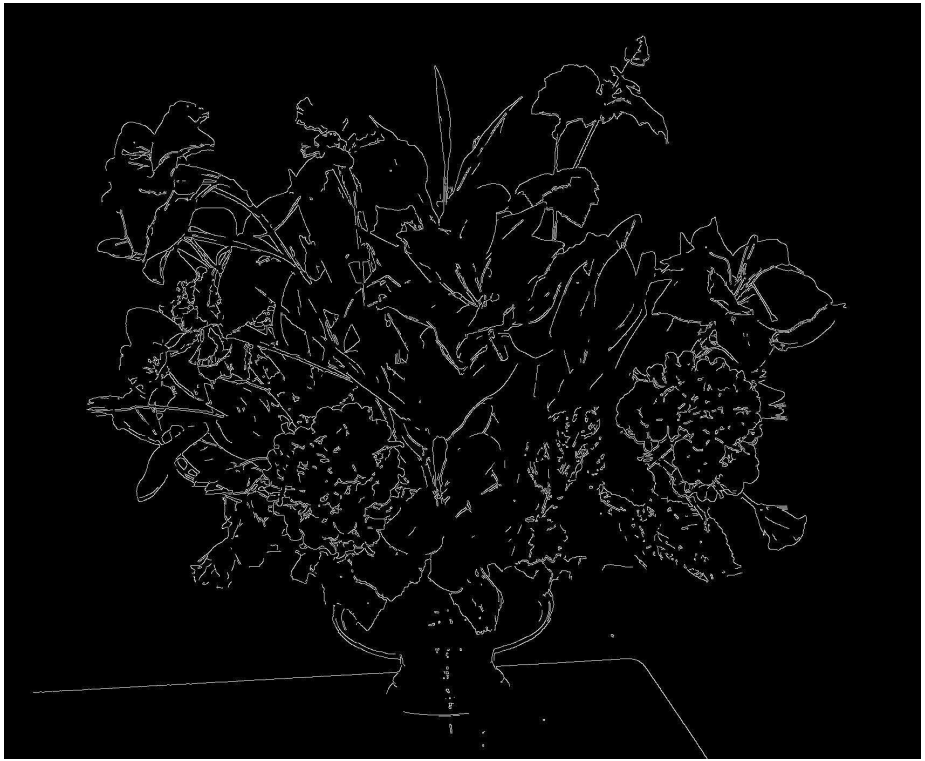
Final Hardware





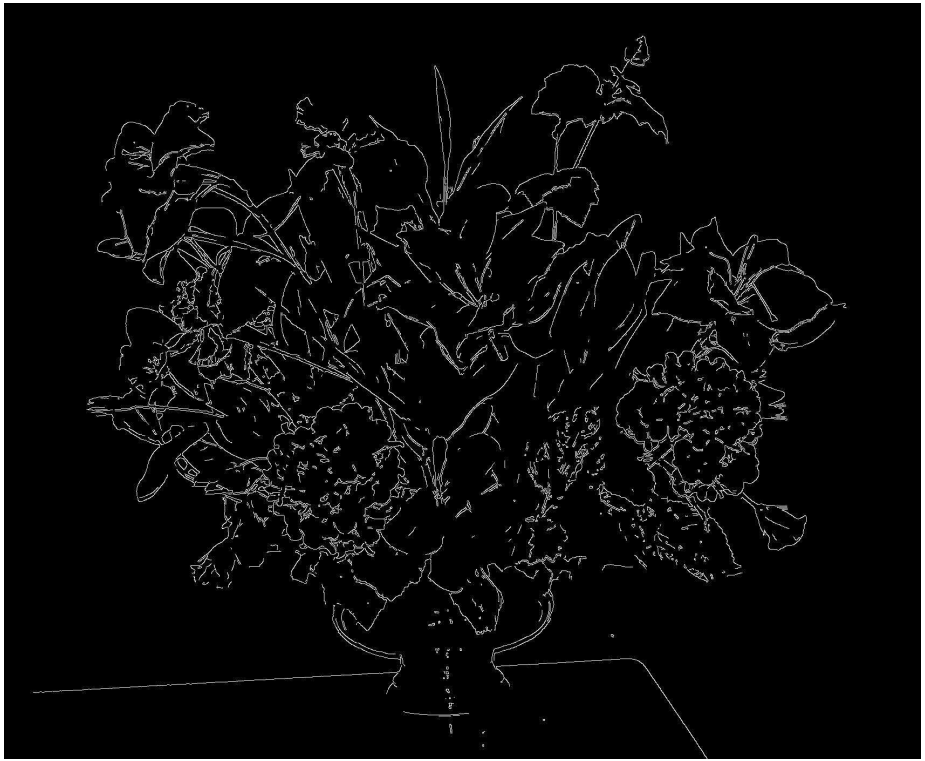
Problem: difficult to distinguish depth discontinuities and edges

Canny Edge Detector





Canny Edge Detector



Multi-flash Edge Detection



Multi-flash Edge Detection

Four images: up, down, left, right flashes

each flash image:

normalize by total grayscale mean

divide by max grayscale isolate shadows

sobel filter on shadow images

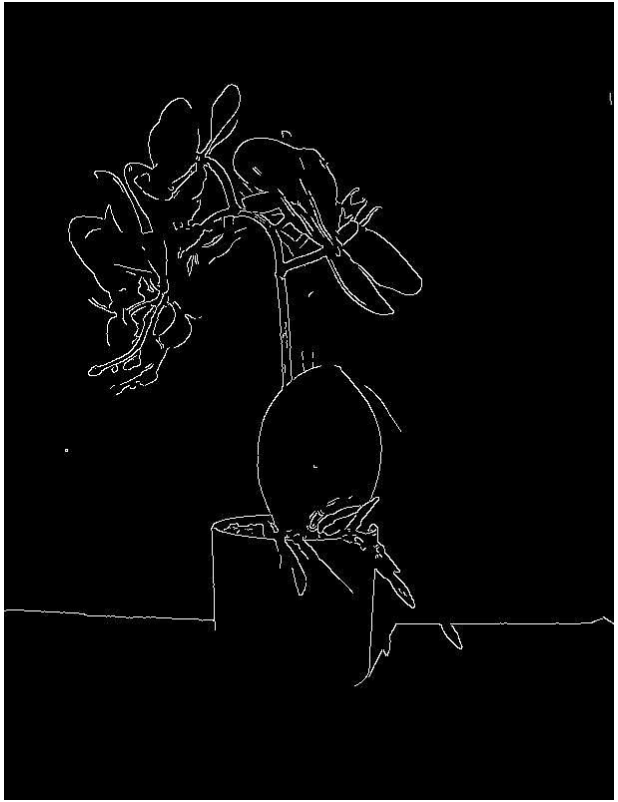
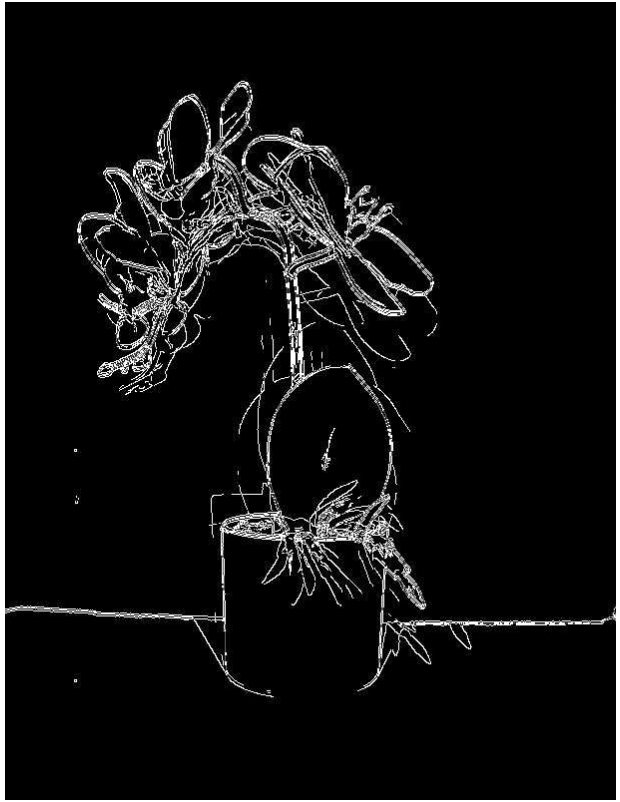
hysteresis filter on silhouettes



Detecting Silhouettes / Simple Edge Map





Applications

Shadow Removal



Simple composite of all non-shadowed regions from every image (taking the max RGB value from each image)

Stylized Edge Rendering



Using the confidence map of the edge detections (composited from all flash images) to stylize the image

De-emphasized texturing



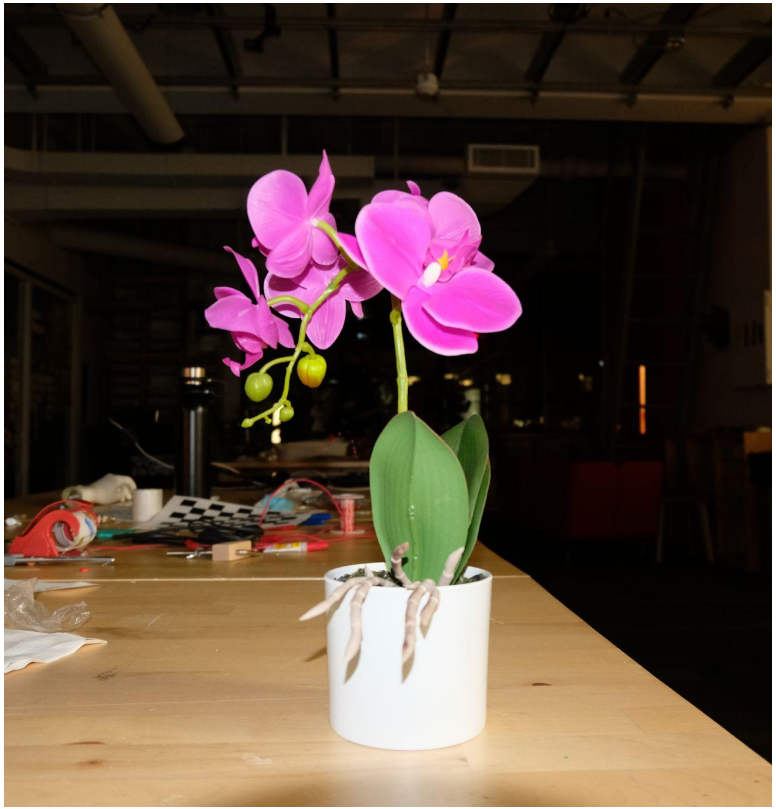
Distinguish foreground from background using depth discontinuities

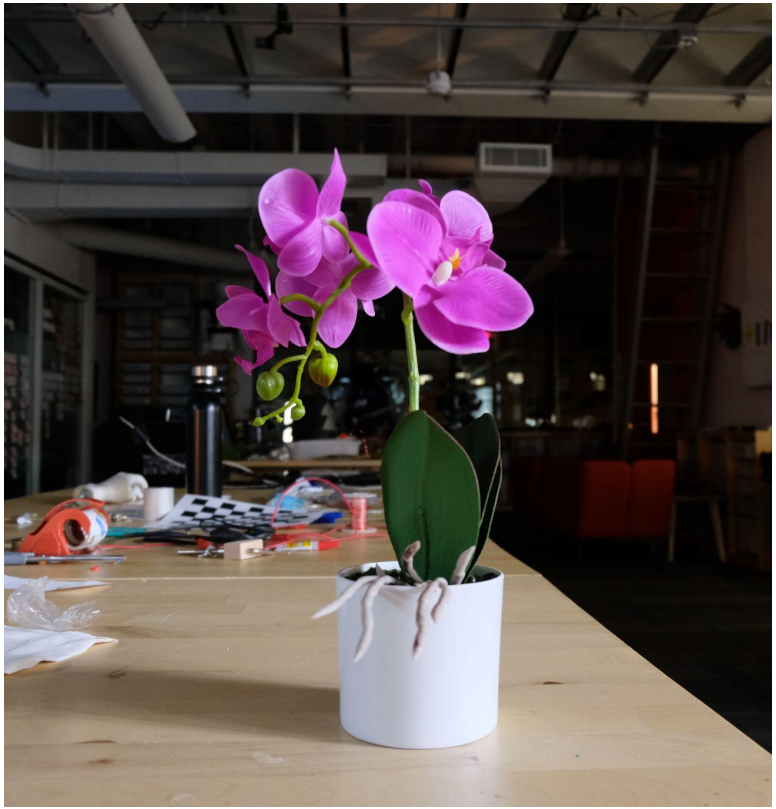
Create a mask using the edge pixels and the texture values near the edge pixels on the “foreground” side

Apply gradient of textures based on euclidean distance

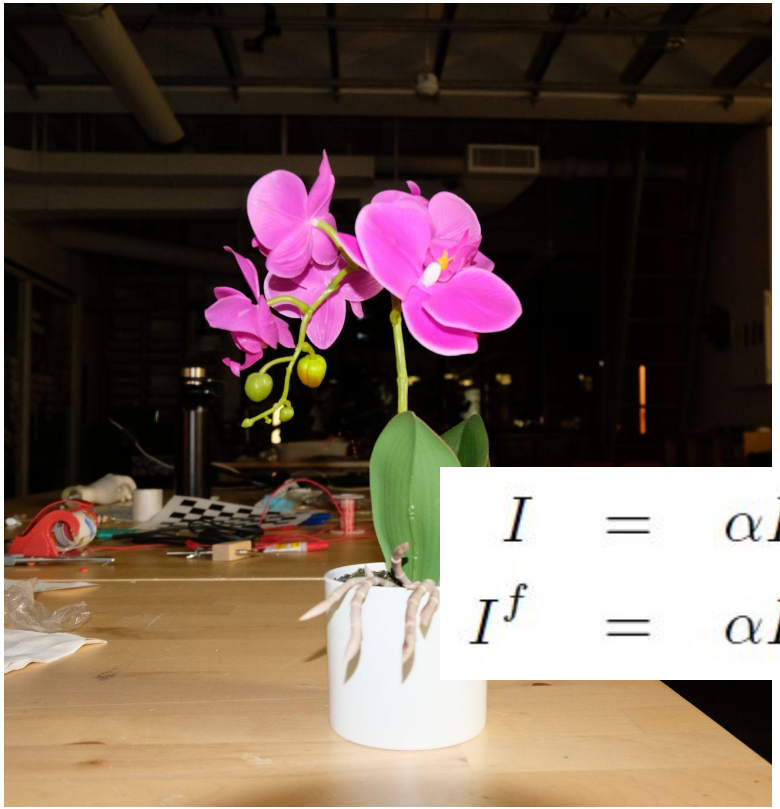
Problem: segmenting foreground and background

Flash Matting





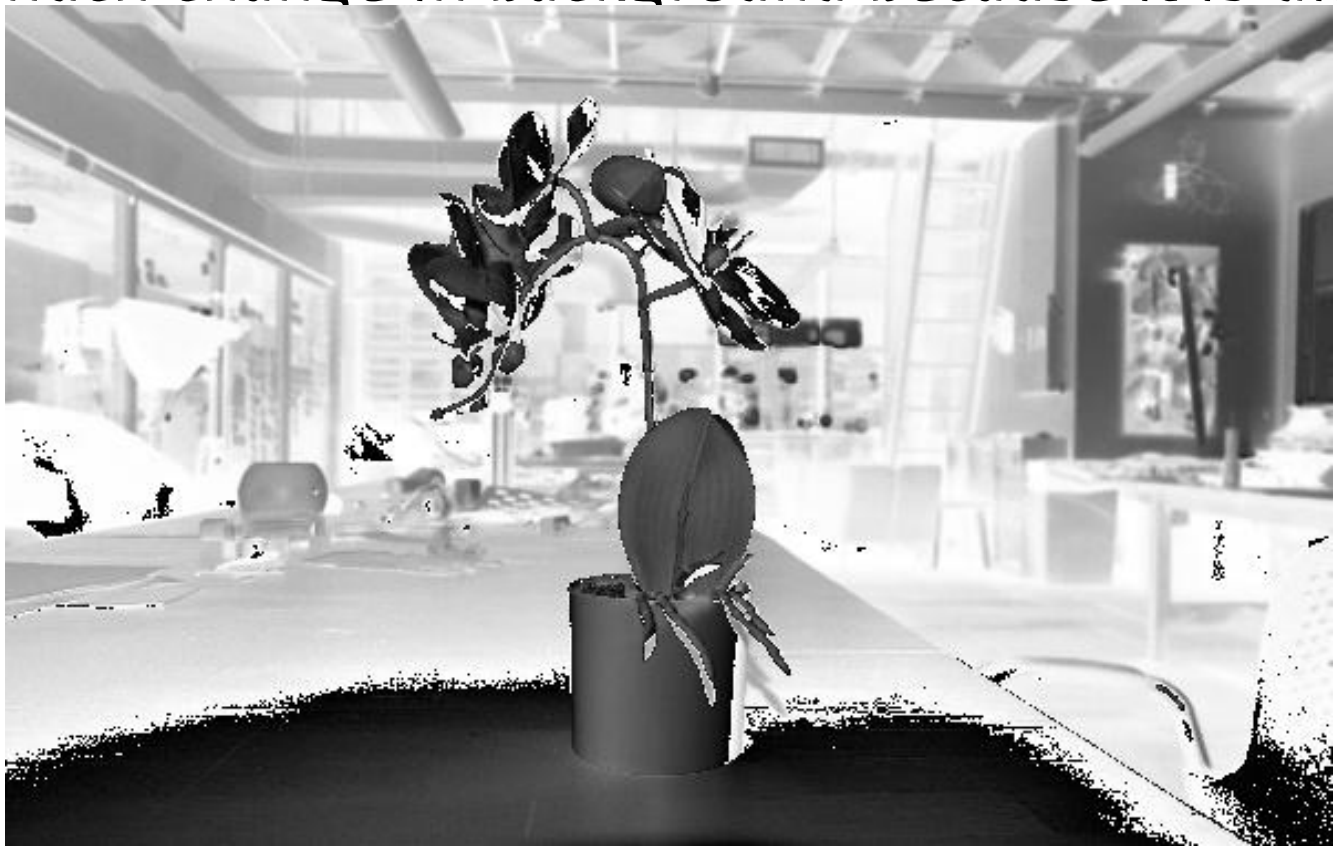
Not much change in background because it is distant





$$
\begin{aligned}
I &= \alpha F + (1-\alpha)B, \\
I^f &= \alpha F^f + (1-\alpha)B^f,
\end{aligned}
$$

Not much change in background because it is distant



Base segmented foreground image



Flash Matting

$$
\begin{aligned}
I &= \alpha F + (1-\alpha)B, \\
I^f &= \alpha F^f + (1-\alpha)B^f \\
& \\
& B^f \approx B. \\
& \\
I^f &= \alpha F^f + (1-\alpha)B.
\end{aligned}
$$

Foreground

$$I' = I^f - I = \alpha(F^f - F) = \alpha F'$$

Bayesian Flash Matting

Generate trimap and apply Bayesian

$$\arg \max_{\alpha, F'} L(\alpha, F' | I')$$

$$= \arg \max_{\alpha, F'} \{ L(I' | \alpha, F') + L(F') + L(\alpha) \}$$

$$L(I' | \alpha, F') = -||I' - \alpha F'|| / \sigma_{I'}^2$$

$$L(F') = -(F' - \overline{F'})^T \Sigma_{F'}^{-1} (F' - \overline{F'})$$



(a) Flash image



(b) Bayesian matting on (a)



(c) No-flash image



(d) Bayesian matting on (c)

Joint Bayesian Flash Matting

$$\arg \max_{\alpha,F,B,F'} L(\alpha, F, B, F'|I, I')$$

$$= \arg \max_{\alpha,F,B,F'} \{L(I|\alpha, F, B) + L(I'|\alpha, F') + L(F) + L(B) + L(F') + L(\alpha)\}$$



(e) Flash-only image



(f) Foreground flash matting



(g) New composition

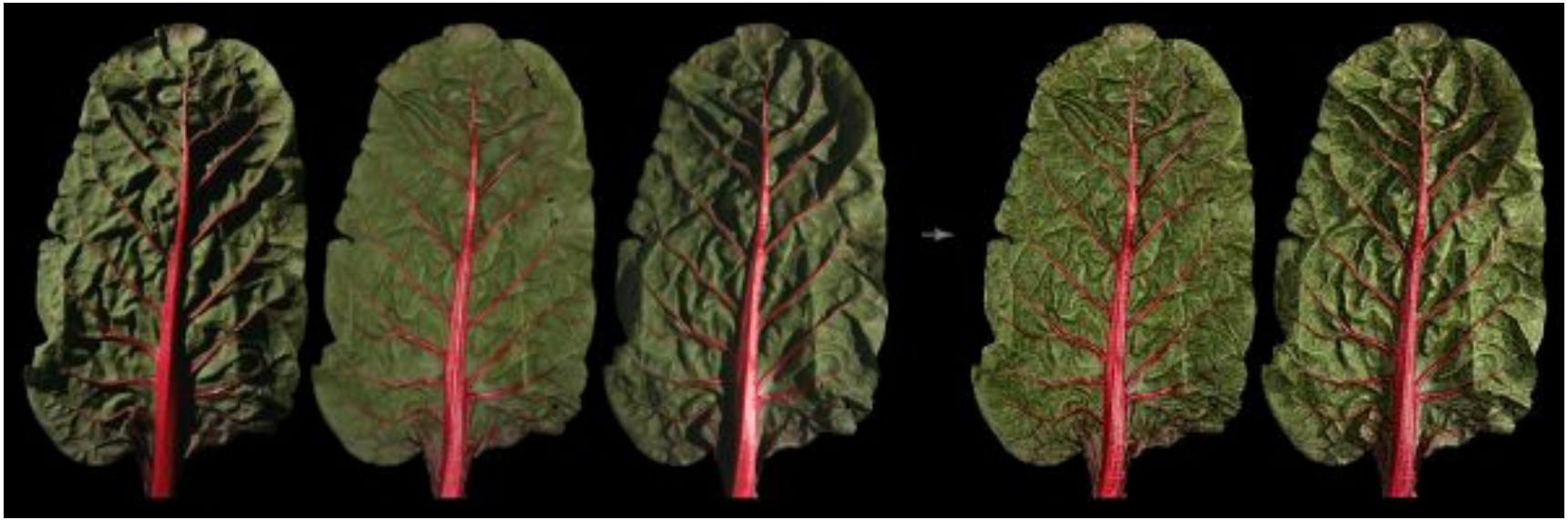


(h) Joint Bayesian flash matting

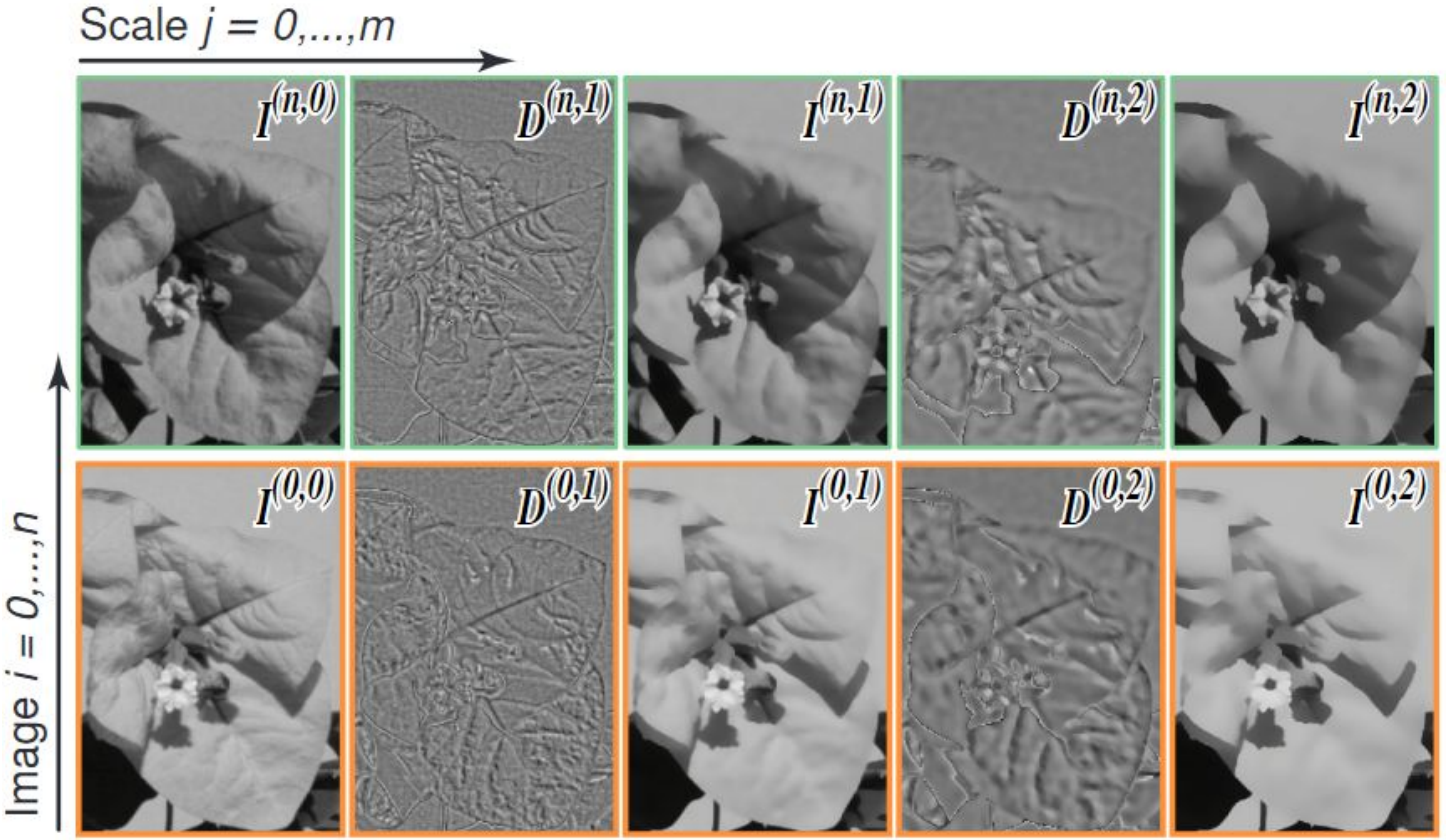
Problem: shading details lost in certain lighting conditions

Detail Enhancement (not implemented yet)

Based on the shadowing introduced from the flash at different angles, different features are highlighted (and shadowed)

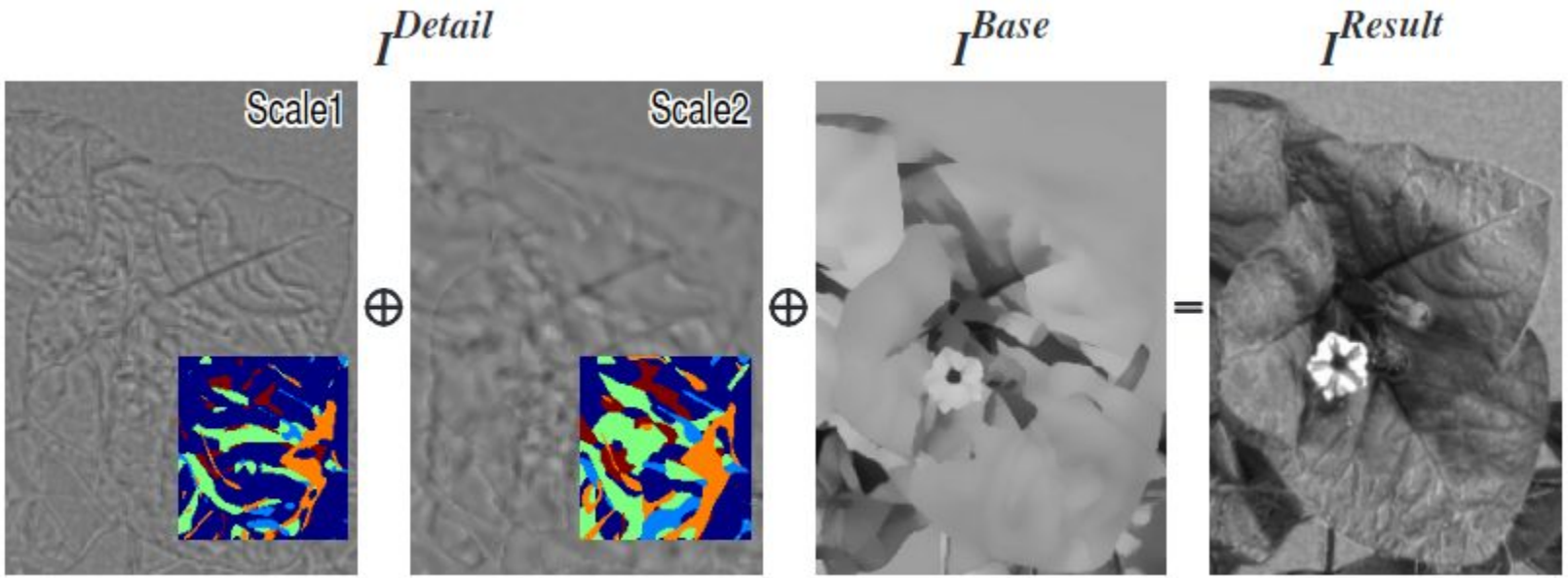


Multiscale Decomposition (per image)



Synthesis

$$I^{Base} = \frac{\sum_{i=1}^{N} \alpha_i I^{(i,m)}}{\sum_{i=1}^{N} \alpha_i}$$



Examples



(a) 1of 5 MLIC Inputs



(b) Result: User Combined



(c) Result: Robust Max.

Examples



Input: 5 MLIC Images

Our Result

Bibliography

- Raskar, Ramesh, Jingyi Yu, and Adrian Ilie. "A non-photorealistic camera: Detecting silhouettes with multi-flash." ACM SIGGRAPH 2003 Technical Sketch (2003).
- Raskar, Ramesh, et al. "Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging." ACM transactions on graphics (TOG) 23.3 (2004): 679-688.
- Sun, Jian, et al. "Flash matting." ACM SIGGRAPH 2006 Papers. 2006. 772-778.
- Fattal, Raanan, Maneesh Agrawala, and Szymon Rusinkiewicz. "Multiscale shape and detail enhancement from multi-light image collections." ACM Trans. Graph. 26.3 (2007): 51.

Event-Based Video Frame Interpolation

An explanation and implementation of TimeLens[1]

Gustavo Silvera
CMU 15-463 Fall 21

Two cameras

Video Camera

- (+) Captures full RGB images (720p, 1080p, etc.)
- (-) Low temporal resolution (24hz, 30hz, 60hz, etc.)



Event Camera

- Captures “events” asynchronously
- Acts similarly to human retina [1]
- (+) Very high temporal resolution (microseconds)
- (+) Very high DR & contrast sensitivity
- (-) Typically lower spatial resolution (~240p)
- (-) $$$



Two cameras

Video Footage:

- still frames



Event Data:

- dense motion

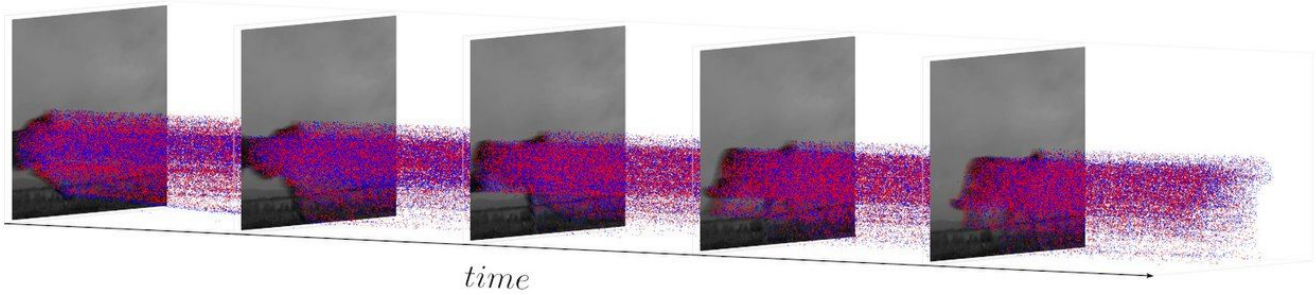
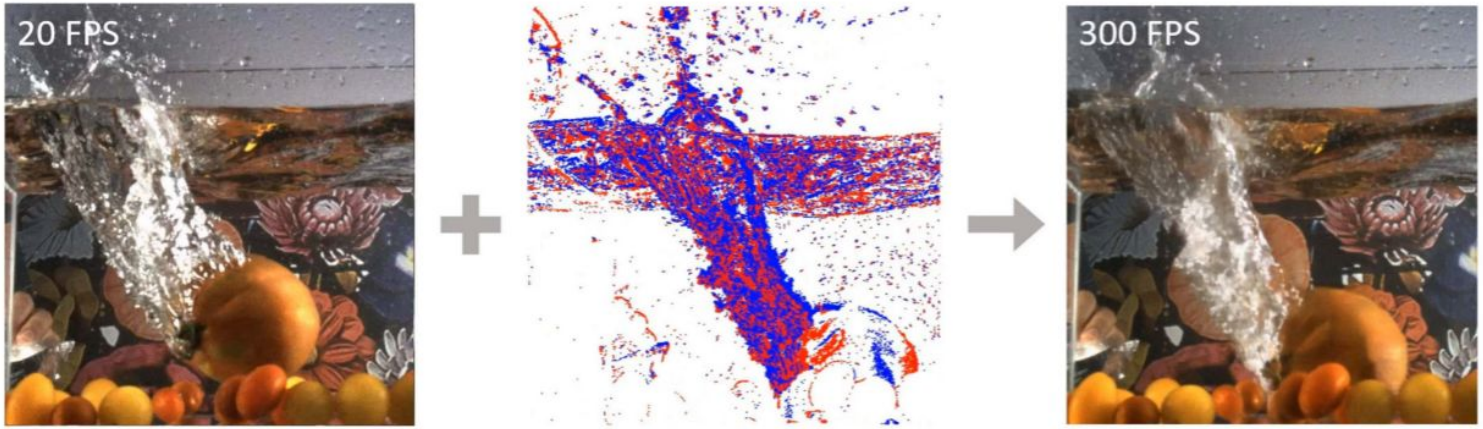


Image Credit: By TimoStoffregen - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=97727937

Combining the two

What if we leverage the high temporal resolution of the event cameras with the high spatial resolution of the video camera to generate a video with the best of both?



Low FPS video + High FPS events => High FPS video

Image Credit: TimeLens Presentation: S. Tulyakov, D. Gehrig, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, D. Scaramuzza, CVPR 2021

How this works (1/2)

What do we have to work with?

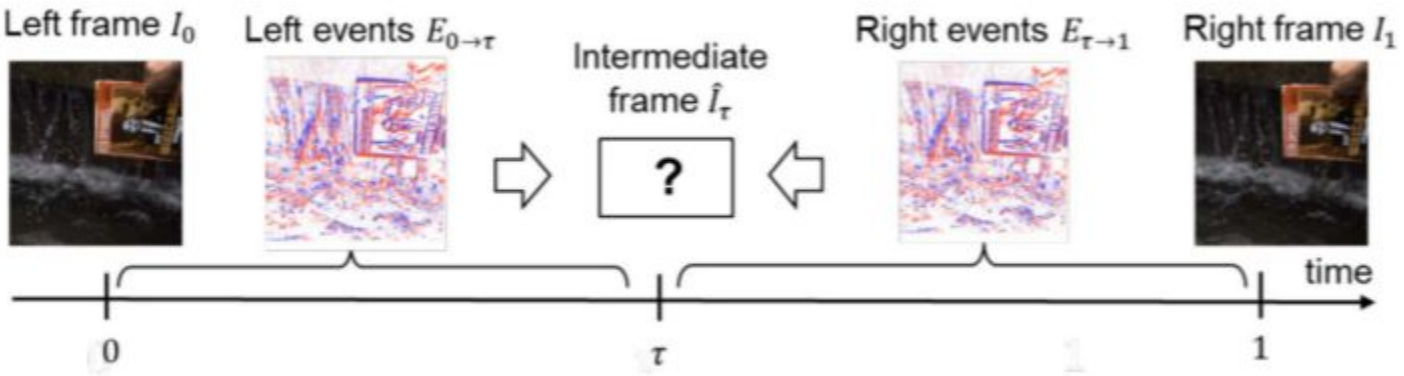


Image Credit: TimeLens Presentation: S. Tulyakov, D. Gehrig, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, D. Scaramuzza, CVPR 2021

How this works (2/2)

How they perform VFI (Video Frame Interpolation)

1. A **warping** operation is done on the **boundary frames** according to the event sequence optical flow
2. The **warping is refined** by computing residual flow (and warping again)
3. Interpolation via **synthesis** is performed (directly fuses keyframe information and event sequences)
4. Optimally combine all three above approaches using an attention-based **averaging** method

These modules are designed to marry the advantages of synthesis VFI and warping VFI

- Synthesis: (+) lighting robustness, (+) sudden changes, (-) texture distortion, (-) noise robustness
- Warping: (-) lighting robustness, (+) non-linear motion, (+) noise robustness

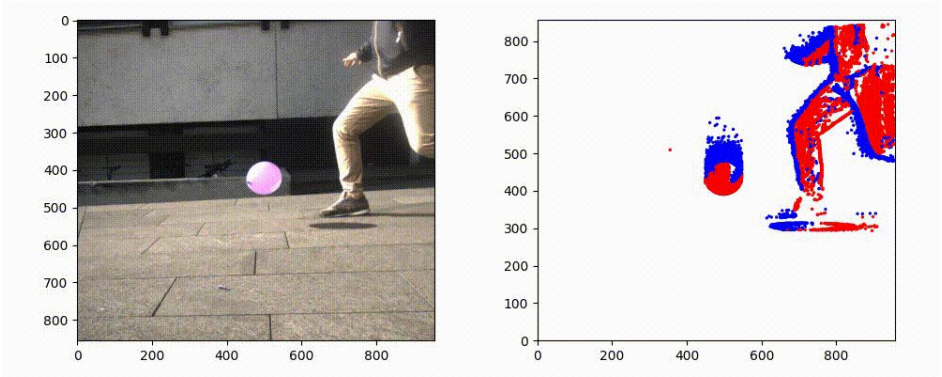
(Read more in Section 2 of their paper[1])

[1] Time Lens: Event-based Video Frame Interpolation. S. Tulyakov, D. Gehrig, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, D. Scaramuzza, CVPR 2021

Example Output

Note that this data was provided by the authors of the paper, but these outputs were generated by my implementation of TimeLens.

Video (30hz)



Events

VFI Video (210hz)



For other cool demos like this, see their webpage (http://rpg.ifi.uzh.ch/TimeLens.html)

My data capture

My setup

Their setup

[Nikon D3500]

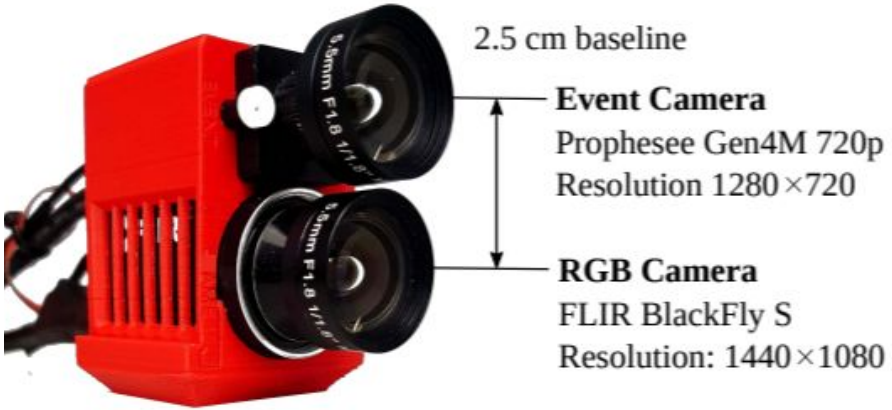
1920x1080@60hz



[DVXplorer Lite]

320x240@5000hz

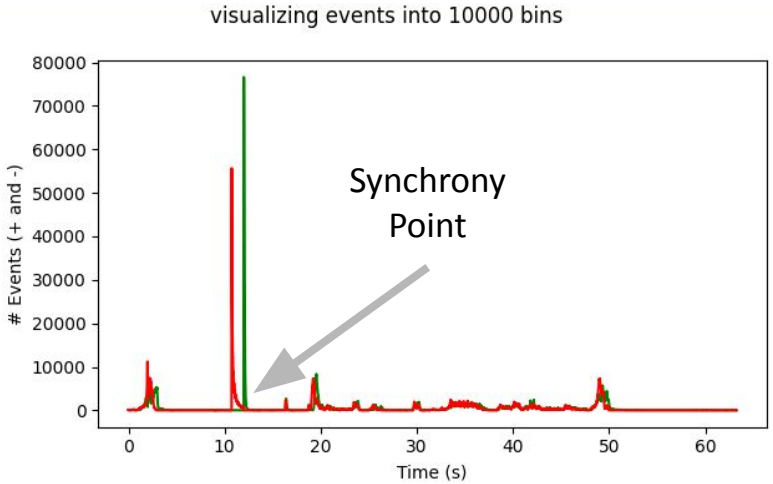


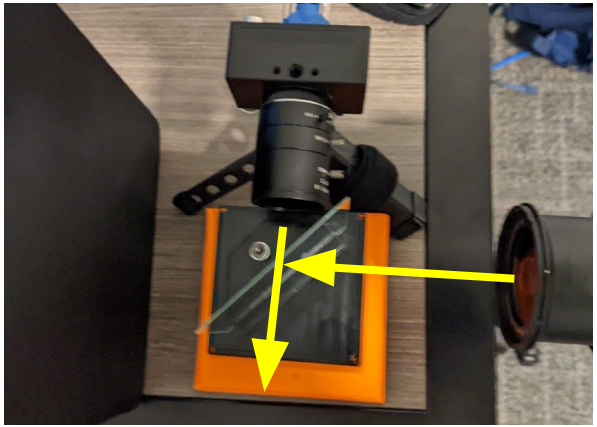


Unforeseen challenges

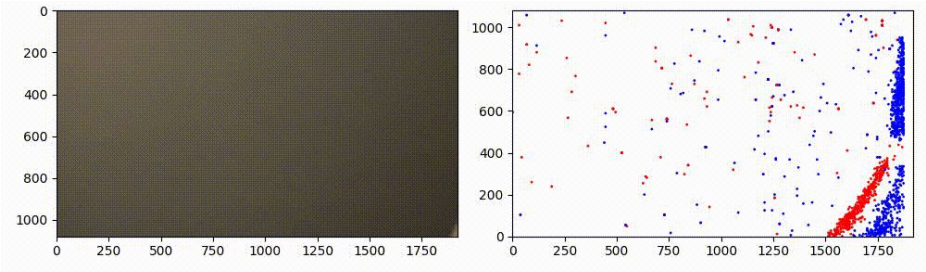
No synchrony between (my) cameras!

- Need to synchronize time (for a common timeline between events and frames)
- Need to synchronize space (both cameras see the same thing)



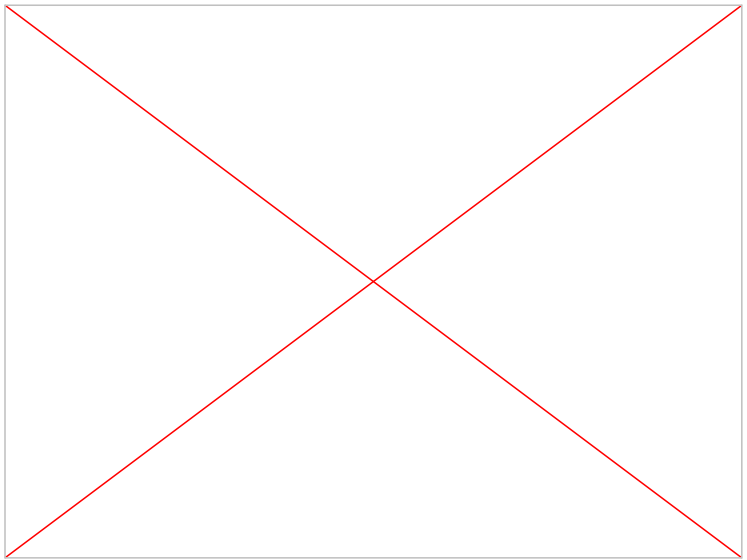


My results (1/3)

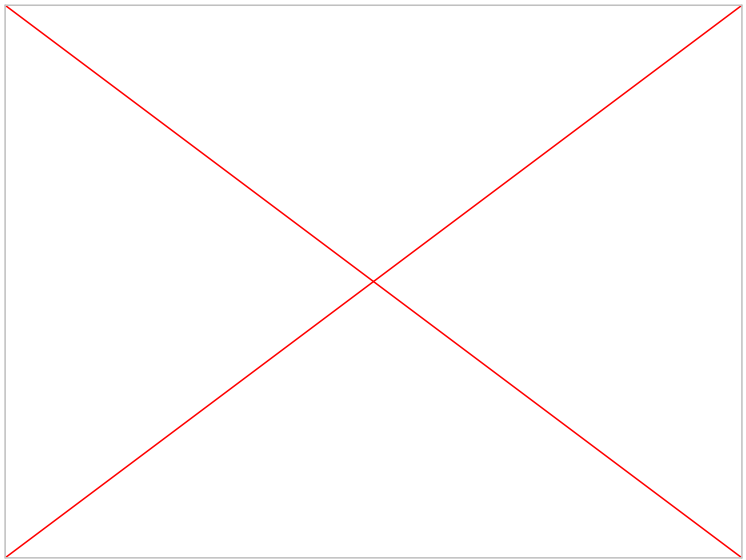


Slowed down (15fps)

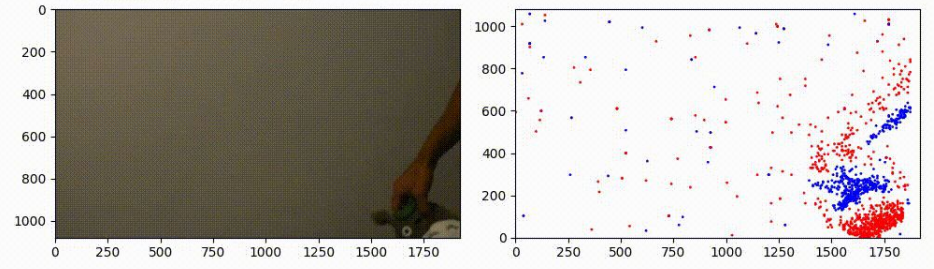
(60fps)



Slowed down

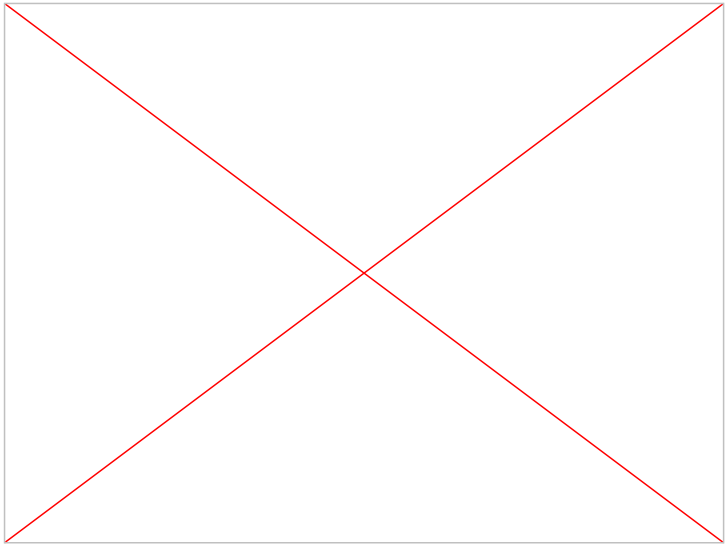


My results (2/3)

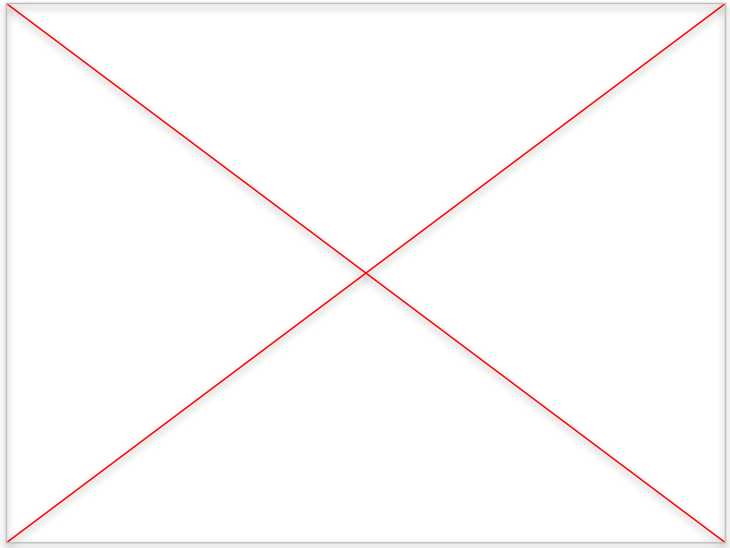


Slowed down (30fps)

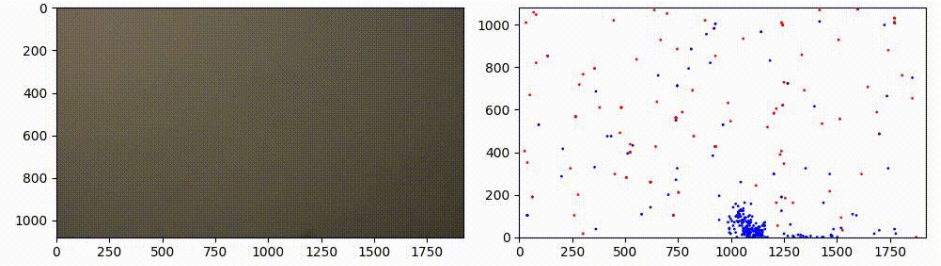
(60fps)



Slowed down



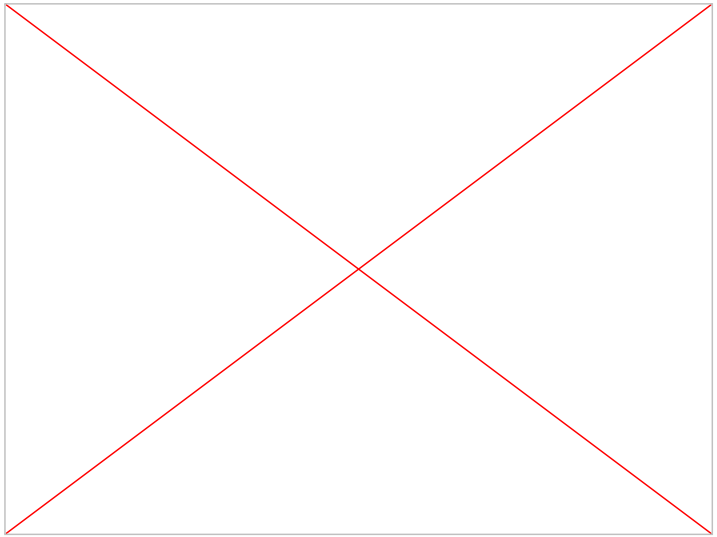
My results (3/3)

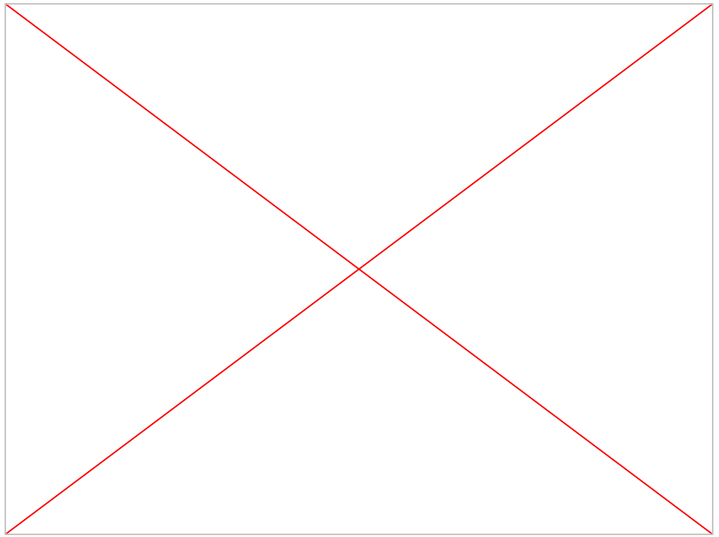


Slowed down (30fps)

(60fps)

Slowed down





Thank you!

Questions?

References:

ETH Zurich Robotics & Perception Group: https://rpg.ifi.uzh.ch

- **Time Lens: Event-based Video Frame Interpolation**: S. Tulyakov*, D. Gehrig*, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, D. Scaramuzza, IEEE Conference on Computer Vision and Pattern Recognition, 2021.

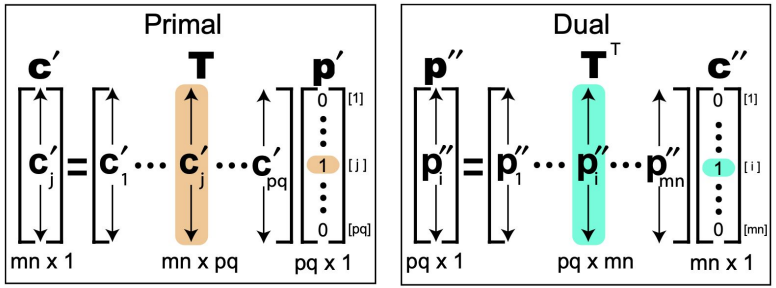
Dual Photography

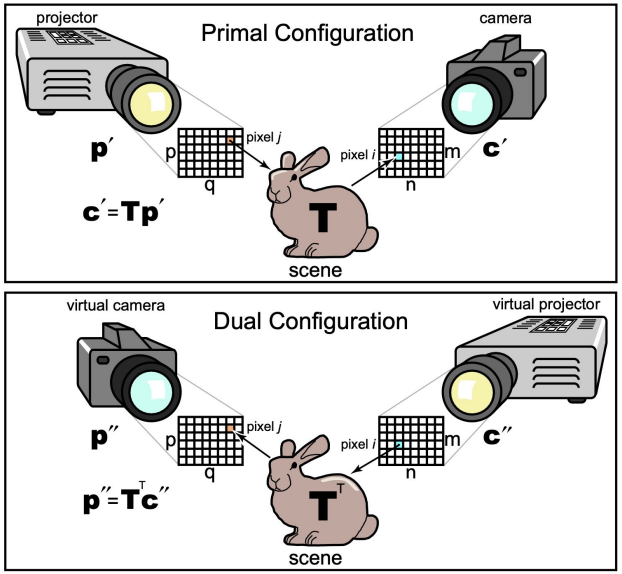
Rachel Tang

Dual Photography

Use **Helmholtz Reciprocity** to produce a virtual image from perspective of the projector without changing setup of scene

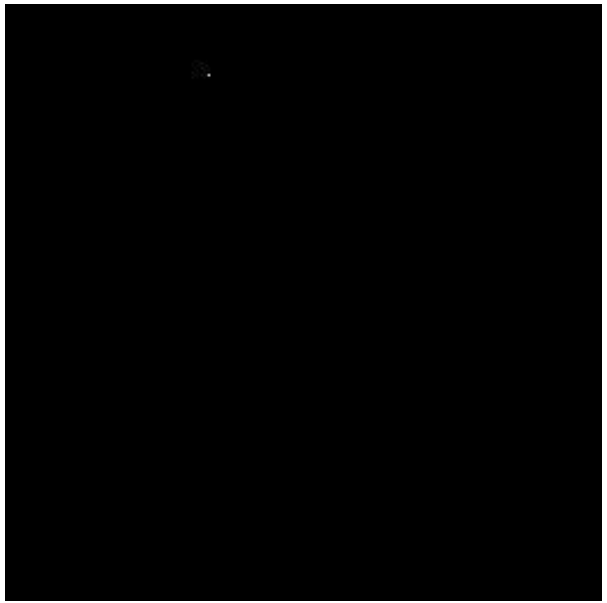
$$f_r(\omega_i \rightarrow \omega_o) = f_r(\omega_o \rightarrow \omega_i)$$



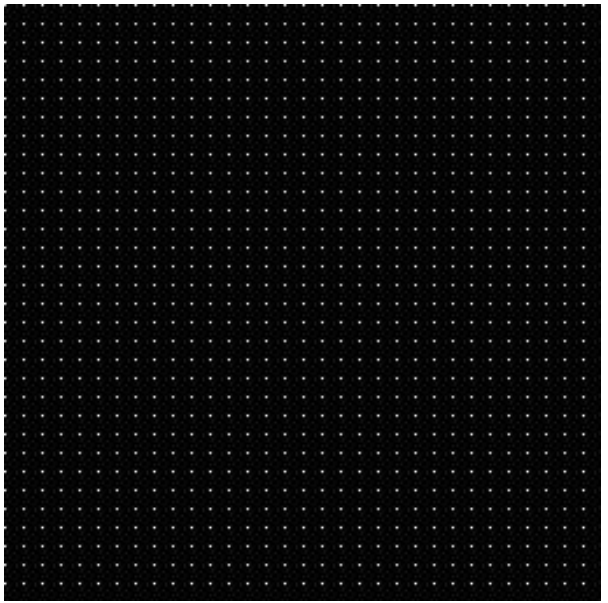


Ways of Capturing Image Dataset

Brute Force Algorithm: Illuminate one pixel on the projector at a time

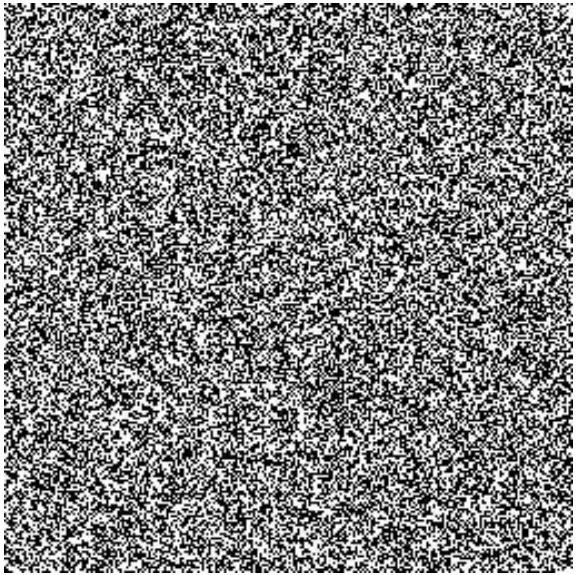


Fixed Scanning: Illuminate several pixels at set interval

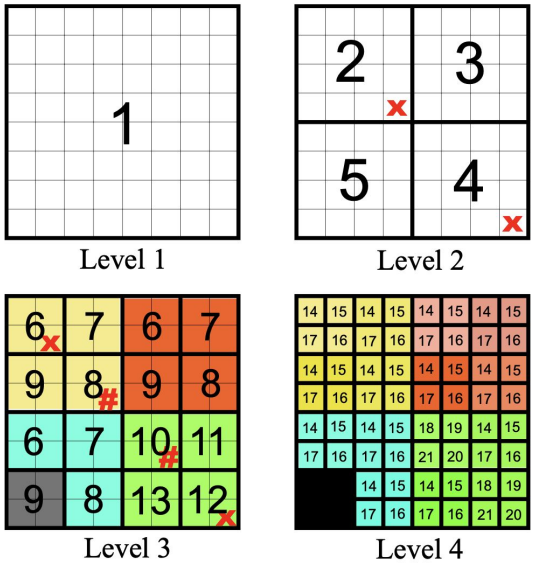


Ways of Capturing Image Dataset

Bernoulli Binary Patterns: illuminate random pixels based on uniform distribution



Adaptive Multiplexed Illumination: recursively calculate which frame a pixel should be illuminated in based on conflicts



Computation of the Matrix **T**

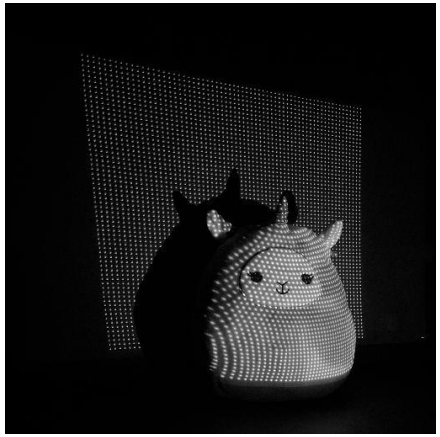
Captured matrices **C** (m*n, k) and **P** (p*q, k) → want to calculate **T** (m*n, p*q)

- $\mathbf{C} = \mathbf{T}\,\mathbf{P}$
- $\mathbf{C}^T = \mathbf{P}^T\,\mathbf{T}^T$
  - This gives us m*n least squares problems to solve to get matrix **T**!
- In the Ax = b least squares problem,
  - $A \rightarrow P^T$
  - $b \rightarrow c_i$ (one row of the matrix **C**)
  - $x \rightarrow t_i$ (one row of the light transport matrix **T**)

Intermediate Results + Challenges

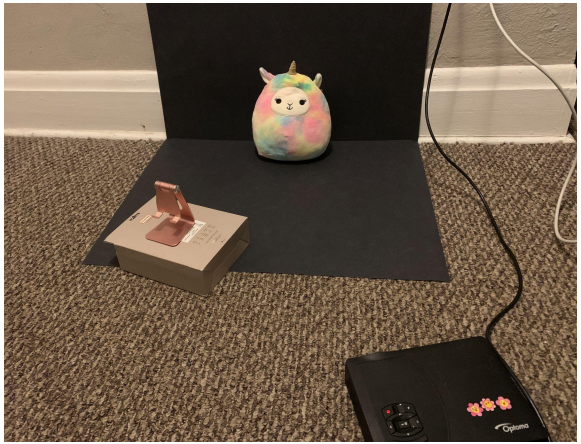
With fixed scanning (debugging images):





Challenges:

- Takes a long time to run the algorithms
- Requires a large amount of memory



References

Sen, Pradeep, and Soheil Darabi. “Compressive Dual Photography.” *Computer Graphics Forum*, vol. 28, no. 2, 2009, pp. 609–618., https://doi.org/10.1111/j.1467-8659.2009.01401.x.

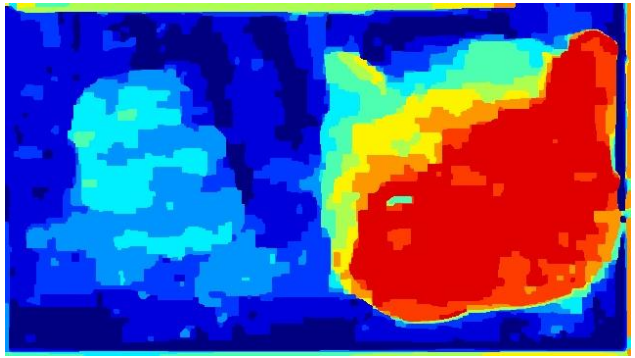
Sen, Pradeep, et al. “Dual Photography.” *ACM SIGGRAPH 2005 Papers on - SIGGRAPH '05*, 2005, https://doi.org/10.1145/1186822.1073257.

Depth Estimation with Two Taps on Your Phone

Chih-Wei Wu | 15663, Fall 2021

Introduction

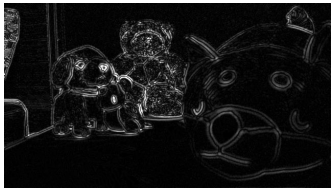
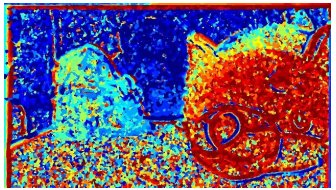
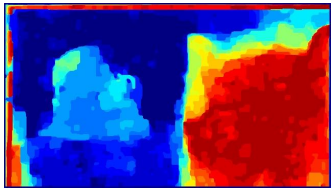




Depth from Focus (*DfF*)

Approach

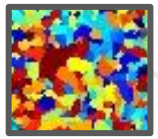
Focal stack alignment → Focus measure → Depth prediction → Depth refinement

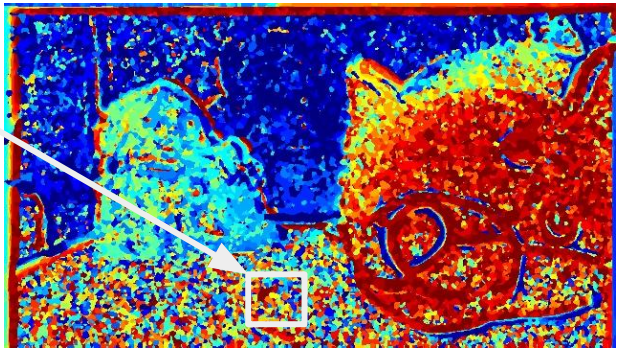





Ref: Suwajanakorn, et al. *"Depth from focus with your mobile phone."* CVPR, 2015.

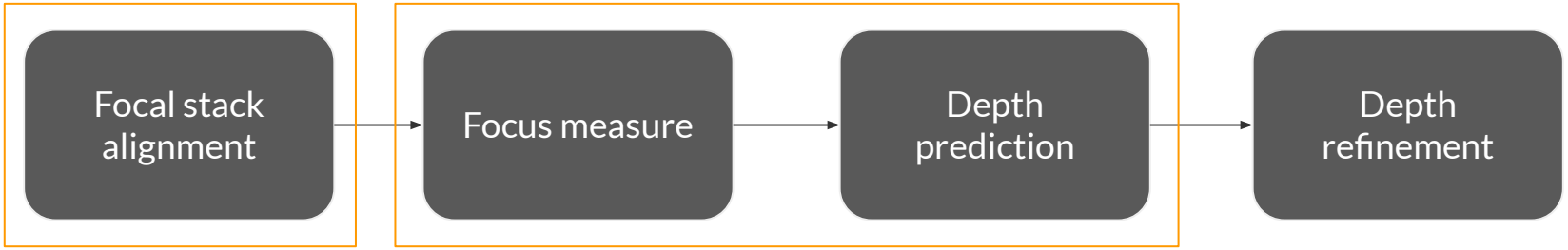
Challenge

No-texture region





Challenge 2:
Noisy depth estimation



Challenge 1:
Camera shake, motion



Dealing with Camera Shake



Input stack

1st improvement



Homography[1]



Convention
Optical Flow[2]



Deep
Optical Flow[3]

Ref 1: Surh, et al. *"Noise robust depth from focus using a ring difference filter."* CVPR, 2017.
Ref 2: Farnebäck, Gunnar. *"Two-frame motion estimation based on polynomial expansion."* SCIA, 2003.
Ref 3: Teed, et al. *"Raft: Recurrent all-pairs field transforms for optical flow."* ECCV, 2020.

Dealing with Camera Shake

Objects are sheared



Homography[1]



Object edges are distorted



Convention Optical Flow[2]



No obvious shear or distortion!



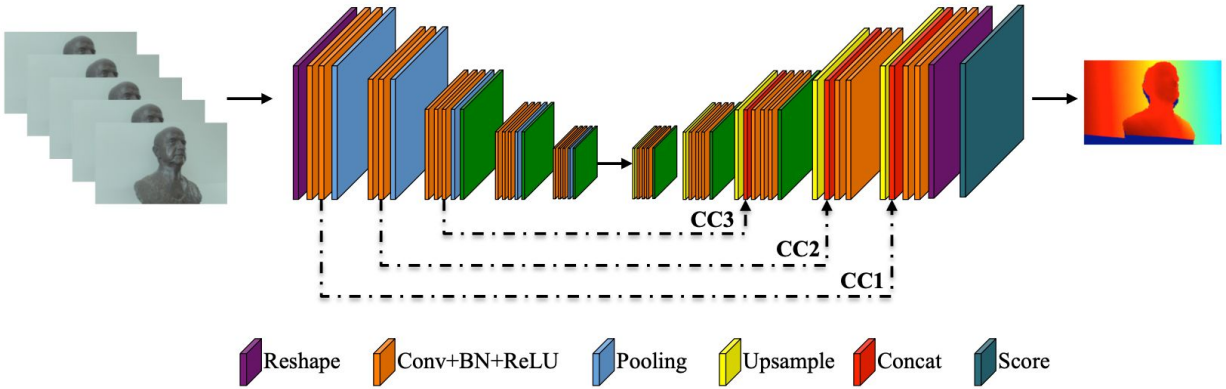
Deep Optical Flow[3]

Ref 1: Surh, et al. *"Noise robust depth from focus using a ring difference filter."* CVPR, 2017.
Ref 2: Farnebäck, Gunnar. *"Two-frame motion estimation based on polynomial expansion."* SCIA, 2003.
Ref 3: Teed, et al. *"Raft: Recurrent all-pairs field transforms for optical flow."* ECCV, 2020.

Dealing with Noisy Depth Estimation

- Handcraft focus measure is not robust enough
- Idea: Train CNN to learn to estimate depth
  - CNN can learn powerful feature as focus measure
  - CNN can learn object shape prior to populate depth for no-texture region

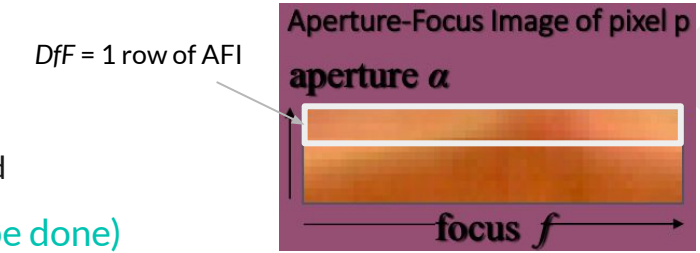
2nd improvement



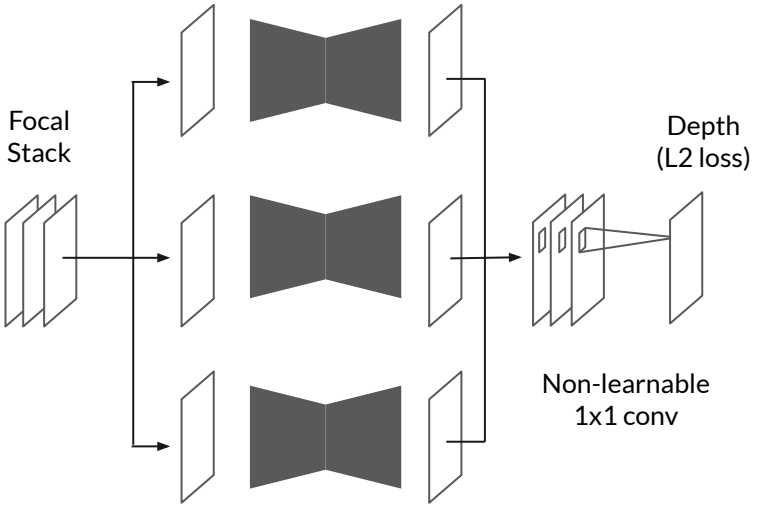
Ref: Hazirbas, et al. "*Deep depth from focus.*" ACCV, 2018.

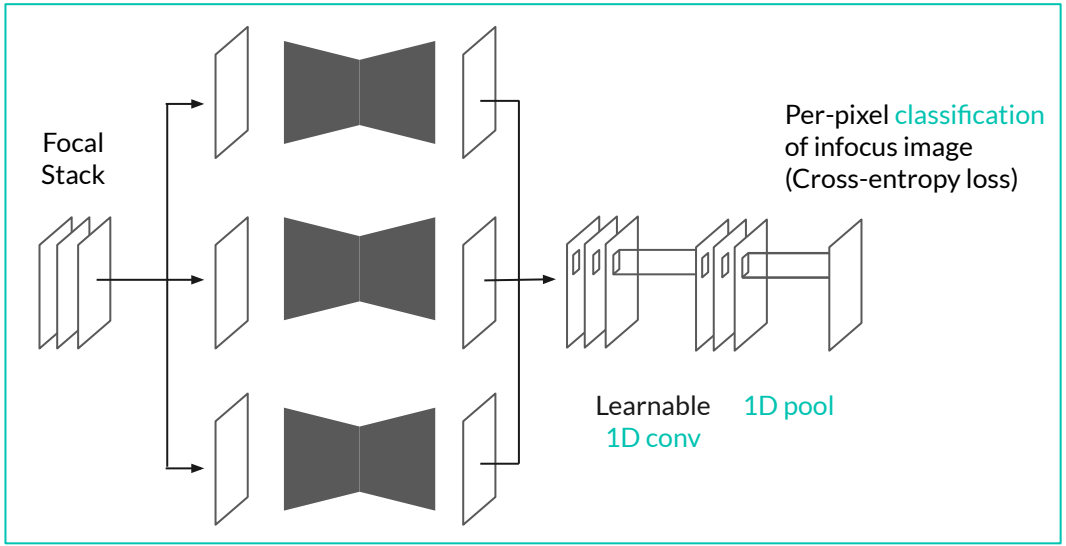
Dealing with Noisy Depth Estimation

- Problem with previous work
  - No feature exchange between stack images
  - Asking the network to predict actual depth is ill-posed



Proposed (To be done)

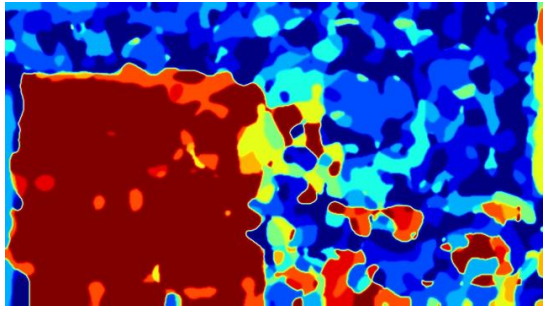




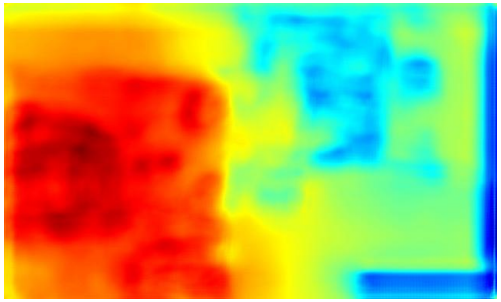
Comparing Depth Estimation Method



Ref: All-in-focus image



LoG + Gaussian Blur



Deep network

Last Secret Weapon

Formulate as MRF multi-label problem,
use graphcut to minimize energy function:

$$E(x) = \underbrace{\sum_{i \in \mathcal{V}} E_i(x_i)}_{\text{Unary term}} + \lambda \underbrace{\sum_{(i,j) \in \mathcal{E}} E_{ij}(x_i, x_j)}_{\text{Pairwise term}}$$

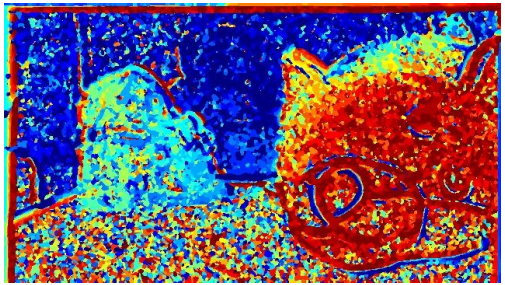
Unary term
Inverse of pixel sharpness

Pairwise term
Inverse of depth smoothness

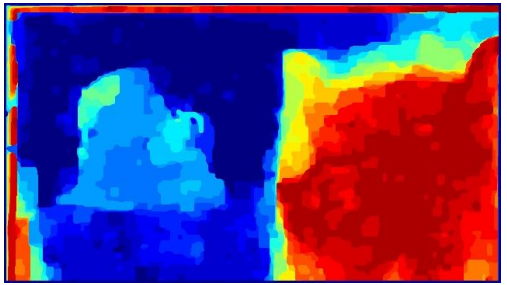
Focal stack alignment → Focus measure → Depth prediction → Depth refinement



Ref: AIF image



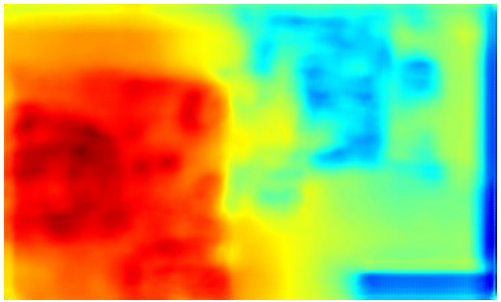
(LoG + Gaussian)



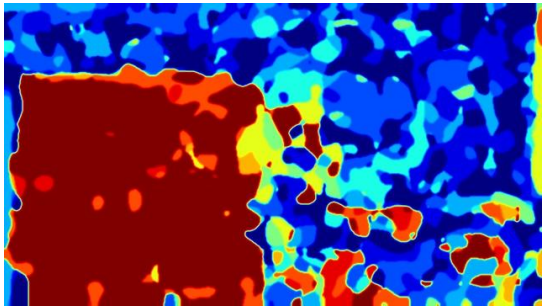
Result



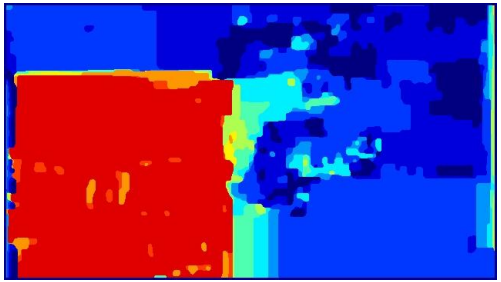
Ref: All-in-focus image



Deep network

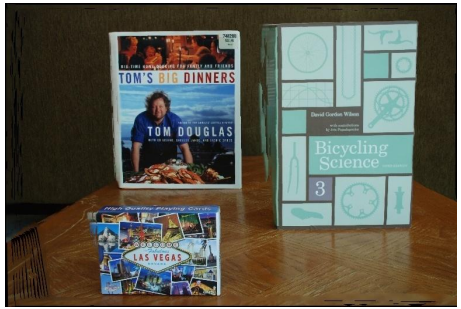


LoG + Gaussian Blur

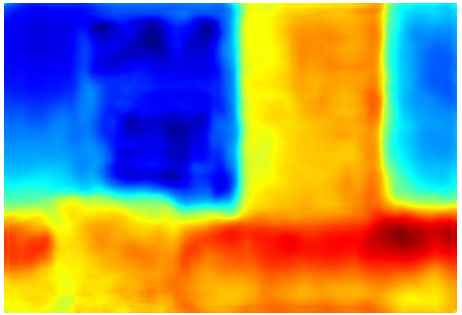


LoG + Gaussian Blur
+ Depth refinement

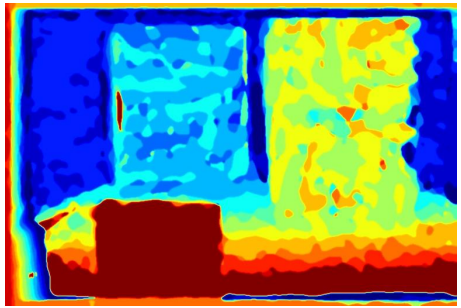
Result



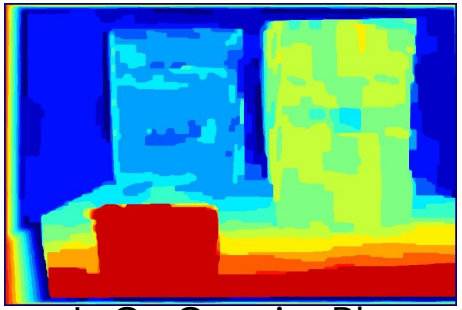
Ref: All-in-focus image



Deep network



LoG + Gaussian Blur

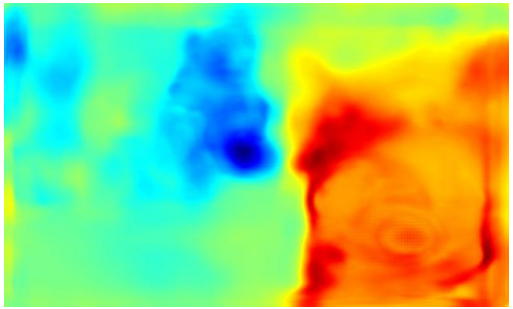


LoG + Gaussian Blur
+ Depth refinement

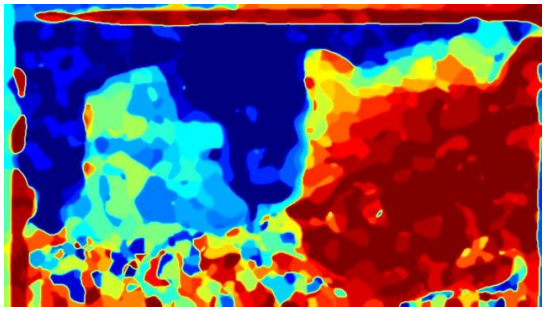
More Result



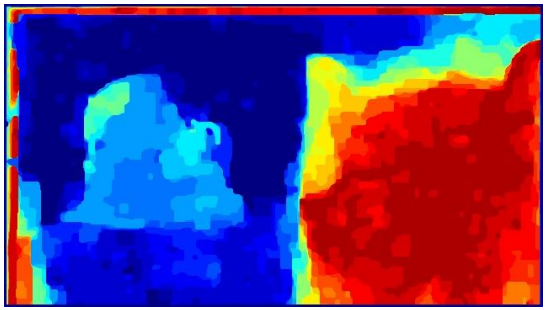
Ref: All-in-focus image



Deep network



LoG + Gaussian Blur

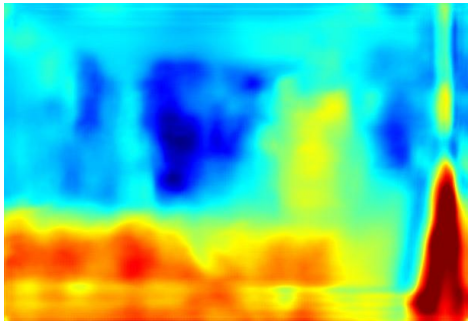


LoG + Gaussian Blur
+ Depth refinement

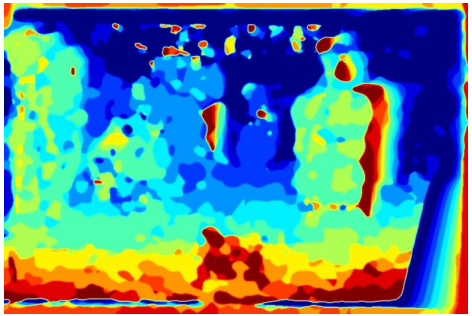
Extreme Case: Super Large Camera Motio



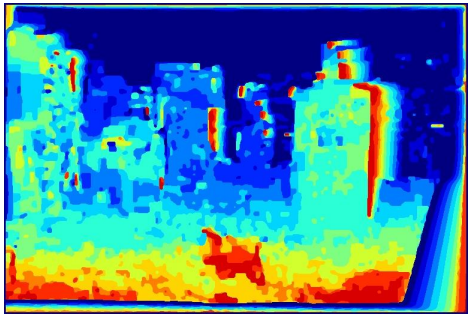
Ref: All-in-focus image



Deep network

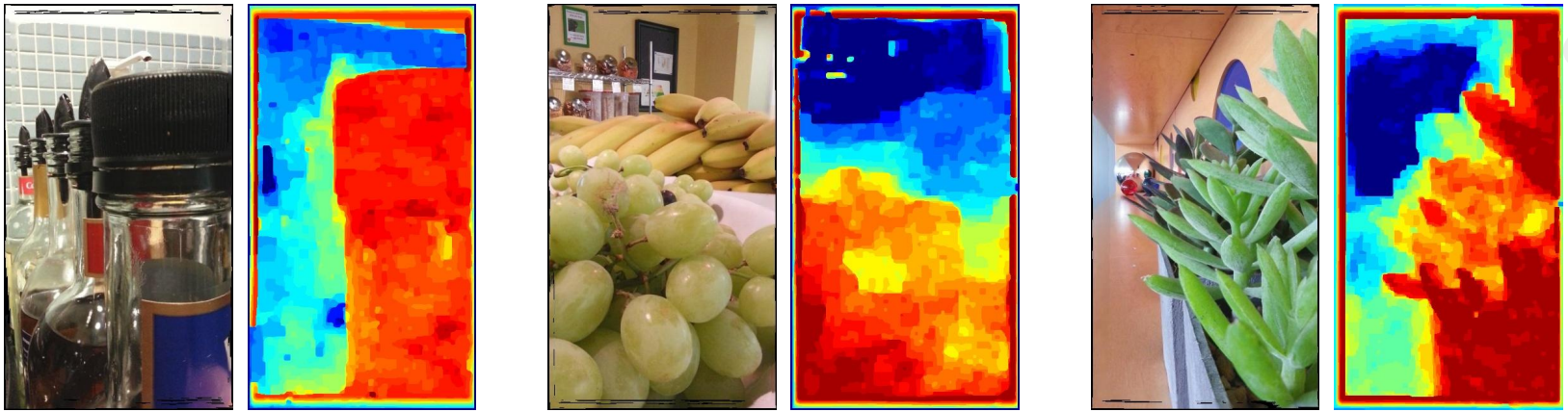


LoG + Gaussian Blur



LoG + Gaussian Blur
+ Depth refinement

Thank you!

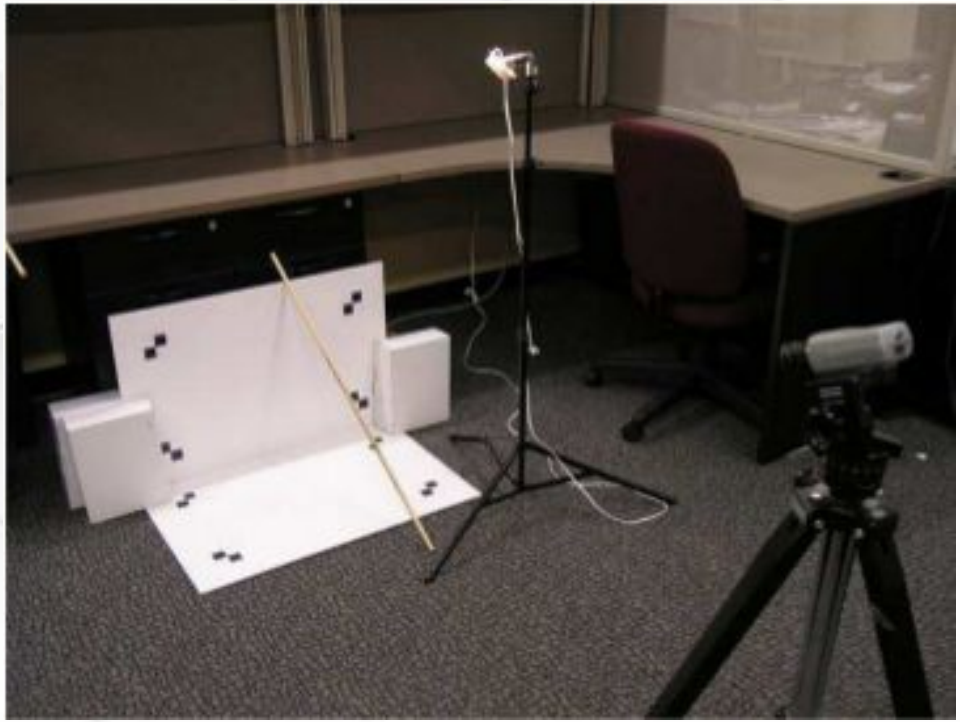


3D Scanning (But not with a Stick)

Daniel Zeng

Assignment 6 - 3D Scanning with a Stick

- Simple setup:



Assignment 6 - 3D Scanning with a Stick

- Wave the stick:



Assignment 6 - 3D Scanning with a Stick

- Frog!



Problems with Stick:

- It’s not very good
- Manually move the stick
- Lots of frames for processing



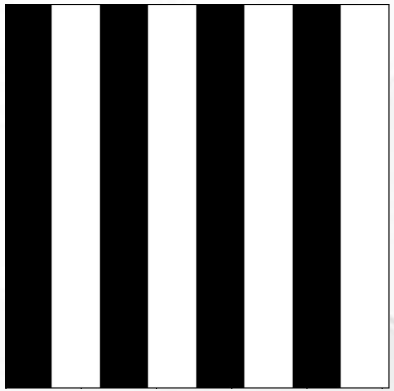


Better method: Structured lighting

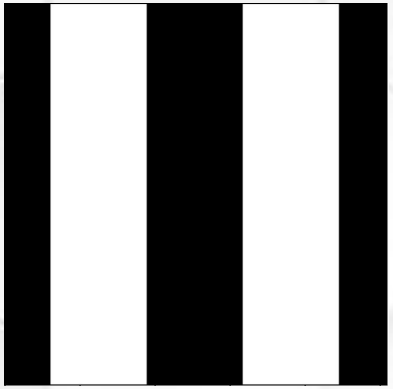
- Use a projector with a specific coding pattern in order to capture the plane
  - Before it was using the shadow from the stick to calculate the plane
- Coding pattern matters a lot!



Discrete Codes:



Binary

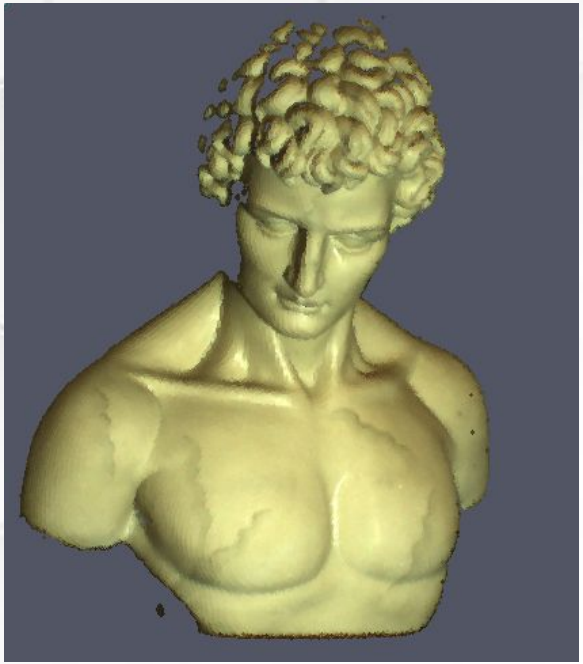


Gray

Example output:

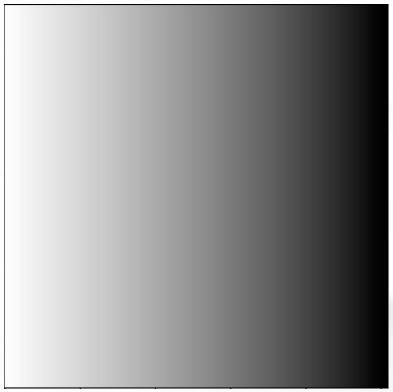


Statue in white light

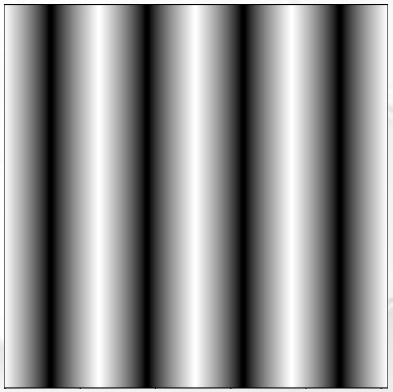


Reconstruction with Gray code

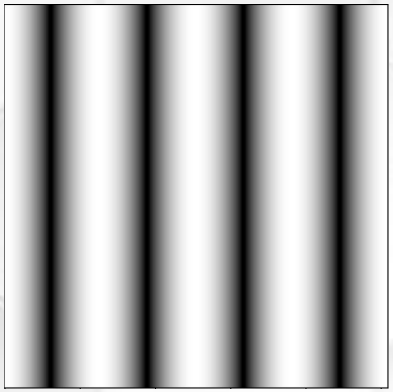
Continuous Codes:



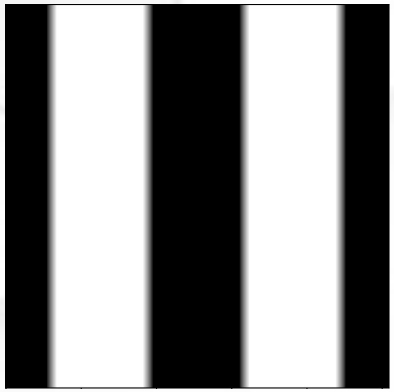
Ramp



Triangle



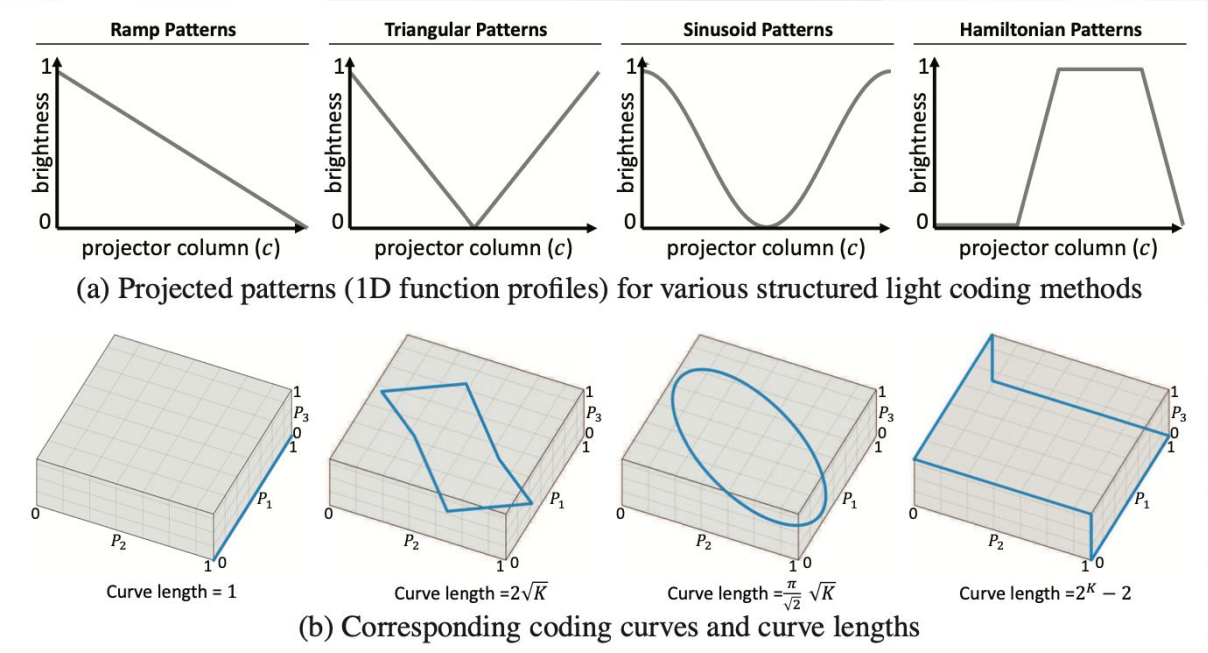
Sinusoid



Hamiltonian

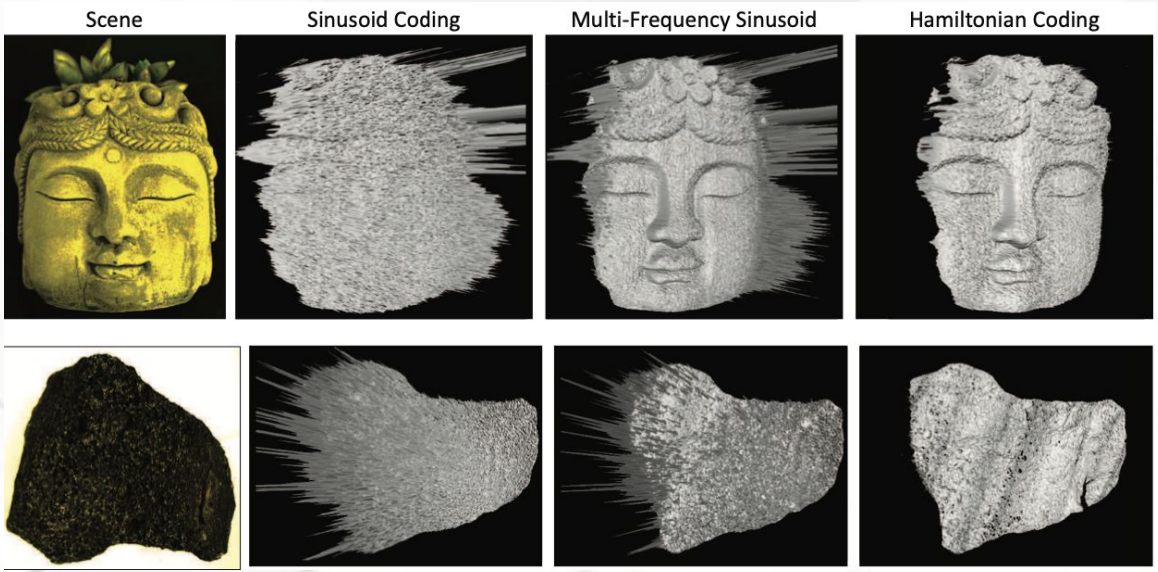
Continuous Codes:

- Curve length is the geometric representation of the code
  - Longer curve = noise more spread out = better code for 3D scanning



Results

- Have not gotten any yet
- Here are some comparisons from the paper:



Acknowledgement

- Mohit Gupta, Nikhil Nakhate; Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 87-102, https://openaccess.thecvf.com/content_ECCV_2018/papers/Mohit_Gupta_A_Geometric_Perspective_ECCV_2018_paper.pdf
  - Hamiltonian Codes
- Build Your Own 3D Scanner course: http://mesh.brown.edu/byo3d/source.html
  - 3D Scanning with structured lighting

Thank you!

Image Colorization Using Optimization

Joyce Zhang

Background

Colorization typically requires:

- Image segmentation + tracking segments over image sequences
- Lots of user input/intervention

Levin's paper "***Colorization using Optimization***" describe a new interactive colorization technique that doesn't precise manual segmentation.



(Levin, 2004)

Algorithm

- Operates on the basic principle that neighboring pixels should have similar colors if their intensities are similar (thus making it edge aware)
- Working in YUV color space
  - Y → intensity
  - U, V → chrominance (encode color)
- compute the weight matrix W that depends on the similarities in the neighbor intensities for a pixel from the original gray-scale image.
- Optimization problem → solve large, sparse system of linear equations; use the user imputed colors as constraints

RGB → YUV

Set up Linear System:

$$J(U) = \sum_{\mathbf{r}} \left(U(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{\mathbf{rs}} U(\mathbf{s}) \right)^2$$

Compute weight matrix **W** of size (imgsize * imgsize)

Weight calculated based neighbors values

$$w_{\mathbf{rs}} \propto e^{-(Y(\mathbf{r}) - Y(\mathbf{s}))^2 / 2\sigma_{\mathbf{r}}^2}$$

Setup **b** based on visual cues provided by the user

Solve linear system of sparse matrix

$$WU = b$$

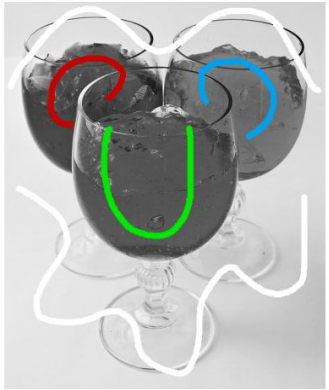
$$U = W^{+}b = (W^T W)^{-1} W^T b$$

Results

Black & White



Color Hints



Colorized



Black & White



Color Hints



Colorized



Not bad!

Idea: Can we use this on non-photorealistic images?

Test: kind of works, but not very well.



Input image filtered with GradientShop, algorithm starts to show strong artifacts.

Black & White



Color Hints



Colorized

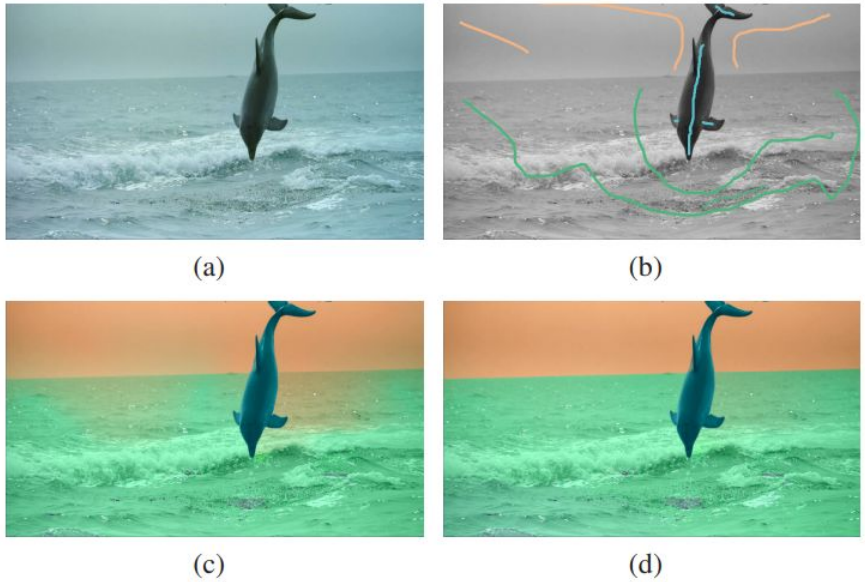


Discussion

- Seems to be very particular about the user input, not sure if it's a problem with my code, or the algorithm itself
- Could be implemented iteratively with Jacobian, which speeds it up
- Potentially can be used as an assistive tool in creative practices. Modern image/video coloration seems to adopt ML-based algorithms instead of user-input-based coloration.

Next Steps

Try to implement the “Colorization filter” from GradientShop, which seems to provide a better algorithm.



Papers referenced

Bhat, Pravin & Zitnick, C. & Cohen, Michael & Curless, Brian. (2010). GradientShop: A gradient-domain optimization framework for image and video filtering. ACM Trans. Graph.. 29.

Levin, A., Lischinski, D., & Weiss, Y. (2004). Colorization using optimization. In ACM SIGGRAPH 2004 Papers (pp. 689-694).

Image Stylization

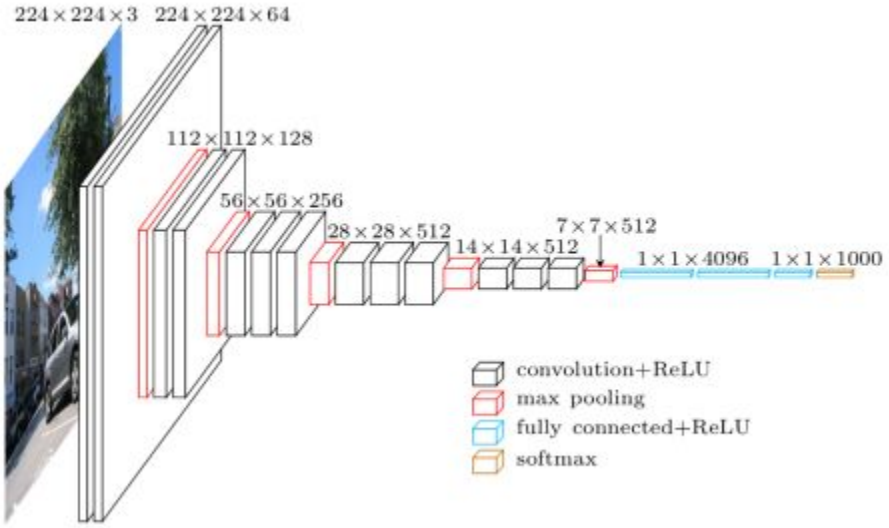
Robin Zheng

Image stylization

(Results from *Image Style Transfer Using Convolutional Neural Network by Gatys*)



VGG-19


224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

Content Representation

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^{l} - P_{ij}^{l}\right)^2 .$$



orginal content

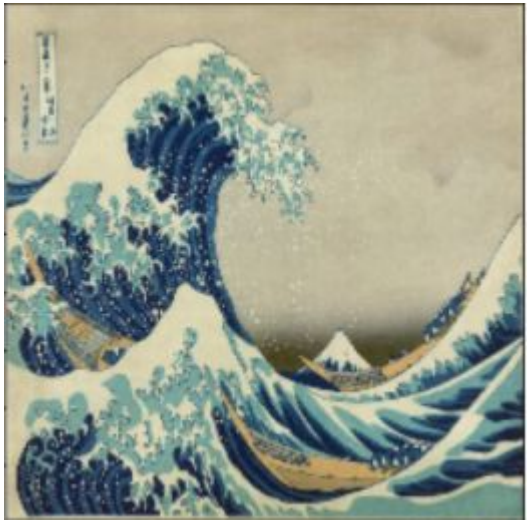


conv1_1

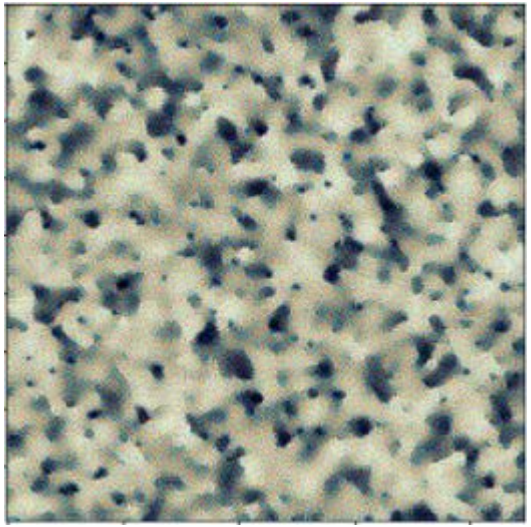
Style representation

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - A_{ij}^l\right)^2$$

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.$$

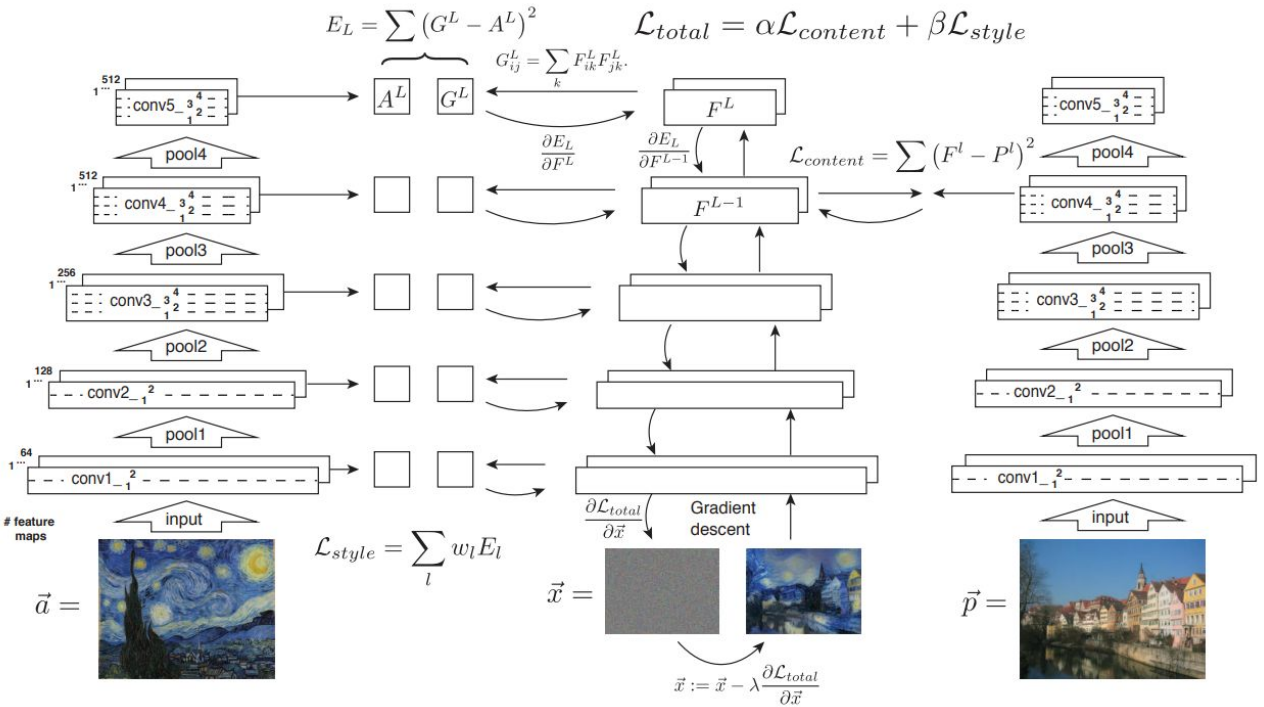


orginal style

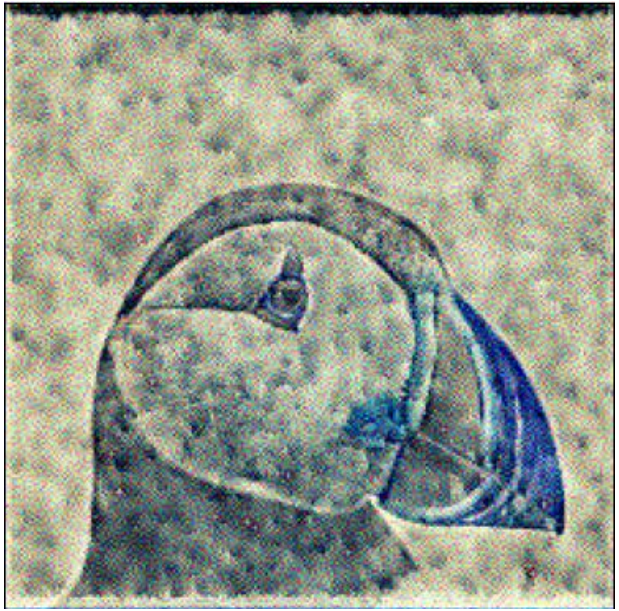


conv1_1

Style Transfer



Some Result (for now)



Reference

Gatys, etc. *Image Style Transfer Using Convolutional Neural Networks*, https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf

Thank you!