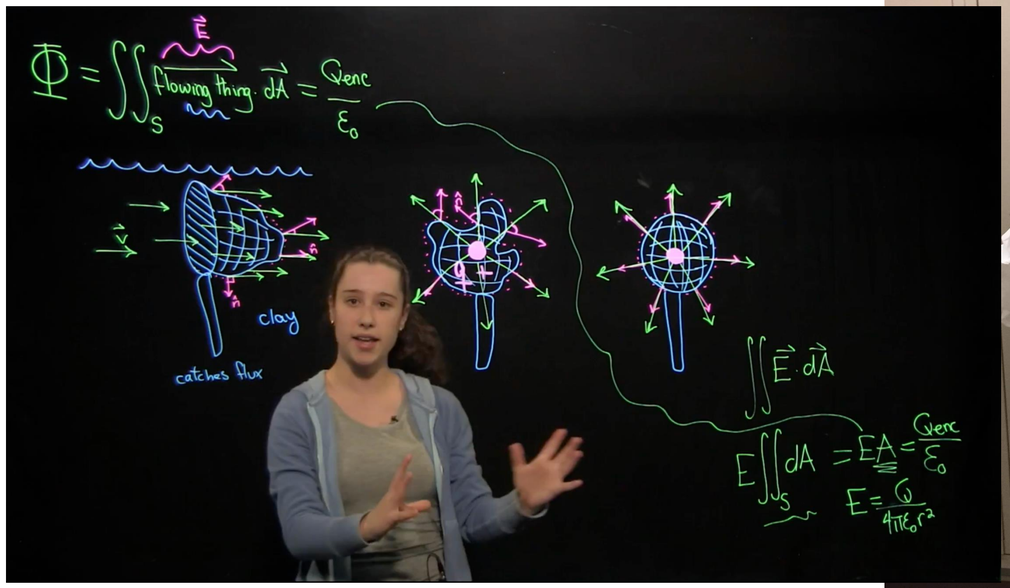
Refraction-based image verification

MEng Research Overview for LincoIn Labs Interview

Sage Simhon

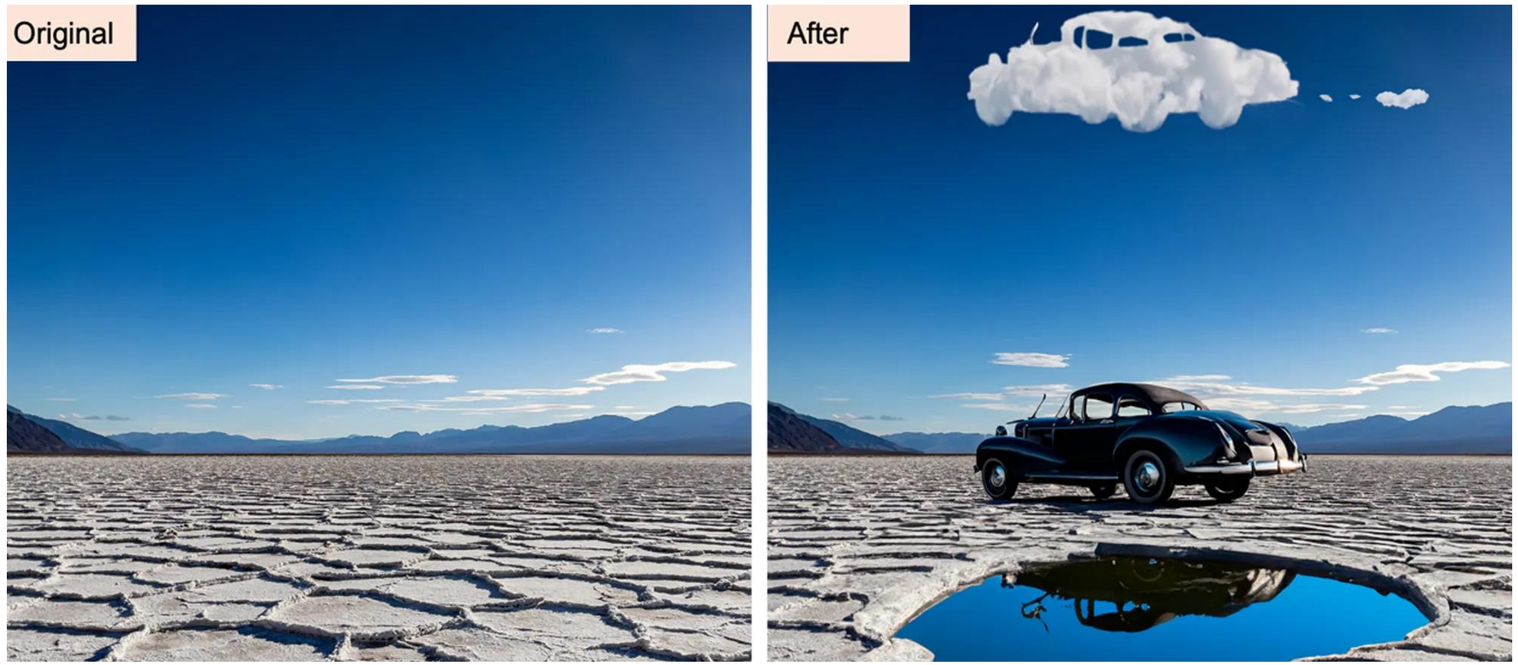
A bit about me

- From Miami, FL
- BSc from MIT in EECS (Course 6-2) (2022)
- MEng from MIT in EECS (2024)
- Diverse interests and background across EE, applied physics, software and AI
  - Some of my favorite coursework:
    - 6.012 (Nanoelectronics and Computing Systems) - how a transistor works from the device physics level!
    - 6.036 (Introduction to Machine Learning) - broad introduction to the field of ML, covering areas including optimization, feature transformations, different types of neural nets, reinforcement learning, & more
      - Became a teaching assistant for the course (first undergrad, then graduate, for 2.5 years)
- Past internships include Apple in silicon + machine learning, Lightmatter
- Went on to join a computer vision/AI lab (Phillip Isola) for my Master's, where I researched refraction-based image verification
- Now, I'm looking to grow as a scientist and engineer, seeking full time roles in applied research settings where I can deepen my skills and contribute to exciting innovative tech



Motivation

AI making image manipulations more advanced and accessible



Adobe photoshop’s new ‘generative fill’ AI tool

Motivation

AI making image manipulations more advanced and accessible

(Video) Facial manipulations are also very feasible with commercial tools

Motivation

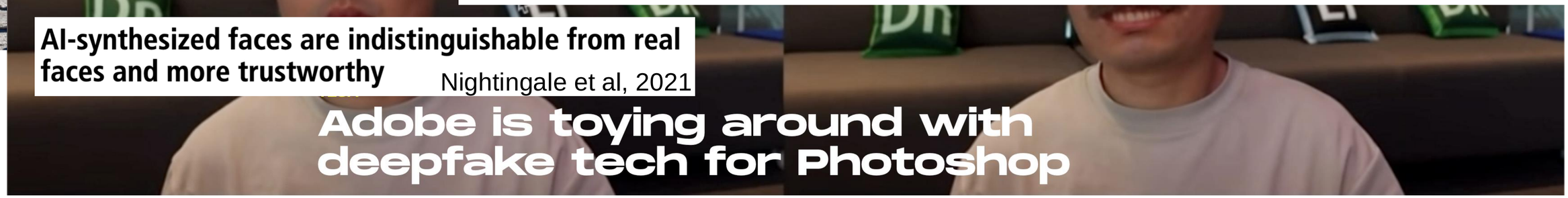
AI making image manipulations more advanced and accessible

Farid et al, 2022

Figure 1. Representative examples of (a) real; (b) GAN-generated; and (c) diffusion-generated faces.

AI-synthesized faces are indistinguishable from real faces and more trustworthy

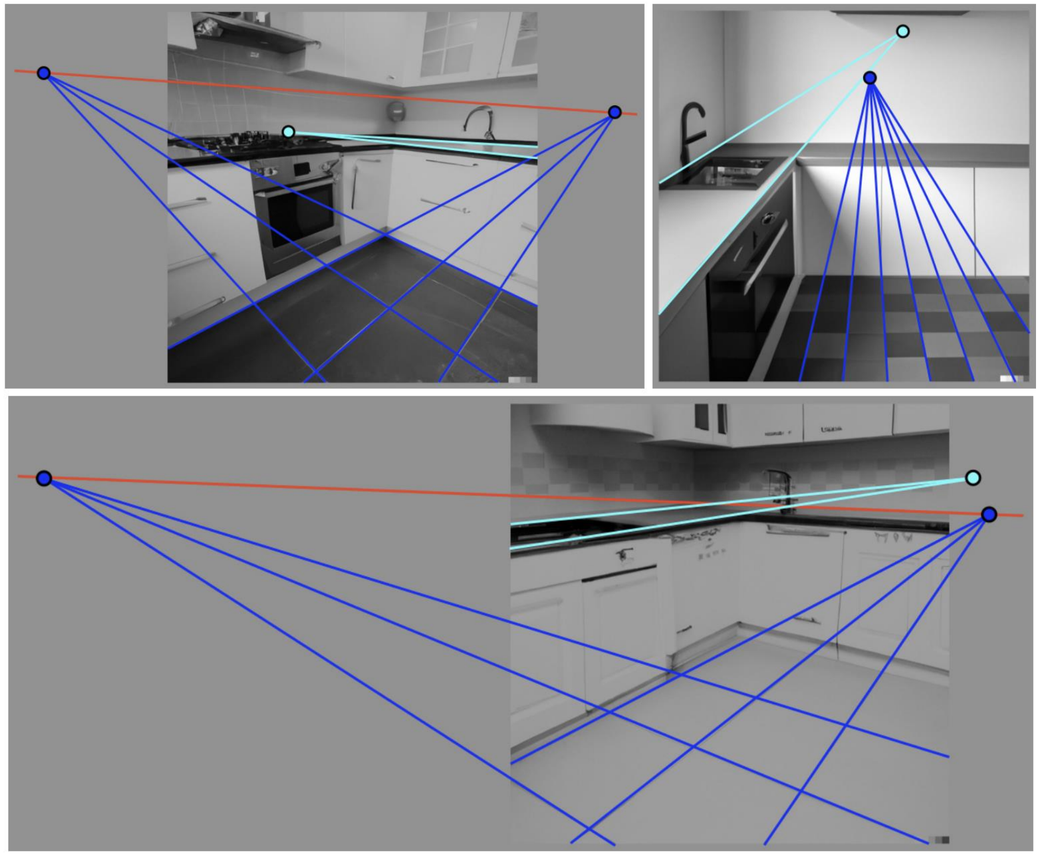
Nightingale et al, 2021



(Video) Facial manipulations are also very feasible with commercial tools

Image verification

Geometric & photometric analyses: shadows, perspective projection, reflections



Perspective (In)consistency of Paint by Text
Farid et al, 2022

ML- and data-driven analyses

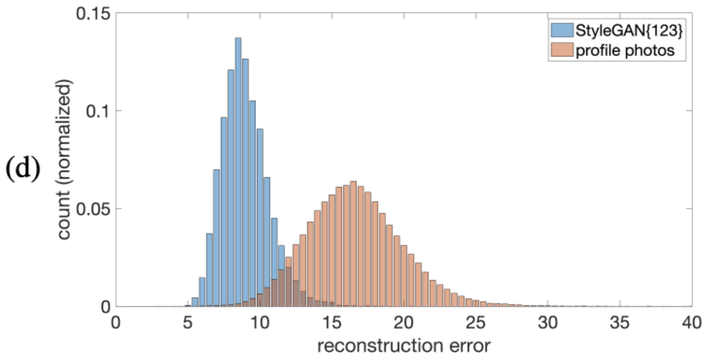


Figure 3. Normalized distributions of image reconstruction error from a learned linear embedding (PCA) (a-c) trained and evaluated separately on three versions of StyleGAN and real profile photos and (d) trained and evaluated on a combination of all three StyleGAN images.

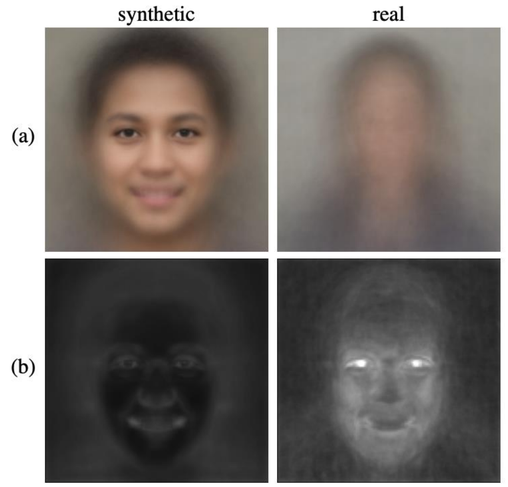
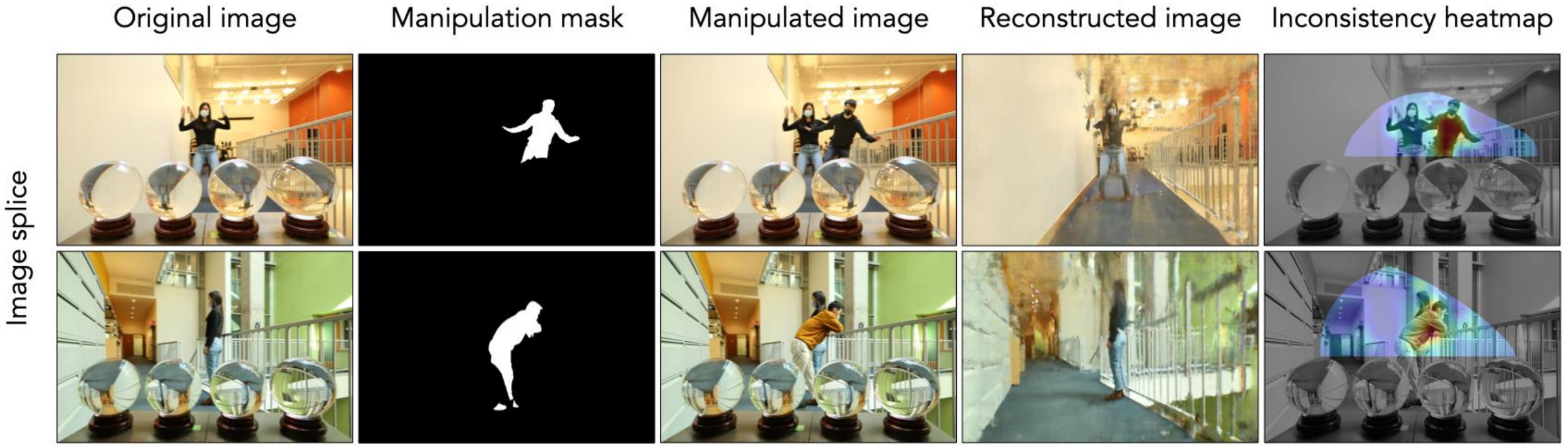


Figure 1. Shown in (a) is the average of 400 StyleGAN2 faces (left) and 400 real profile photos (right), revealing a highly regular synthetic image structure as compared to a highly diverse profile-photo structure. As shown in Figure 2, the StyleGAN2 photos used to create this averaged image were drawn from a diverse demographic pool. Shown in (b) are the average image reconstruction errors (displayed on the same intensity scale) from a learned linear embedding of 10,000 StyleGAN2 faces; this embedding captures the underlying structure of synthetic faces but not profile photos, as seen by the smaller reconstruction error for synthetic faces.

Exposing GAN-Generated Profile Photos from Compact Embeddings
Mundra et al, 2023

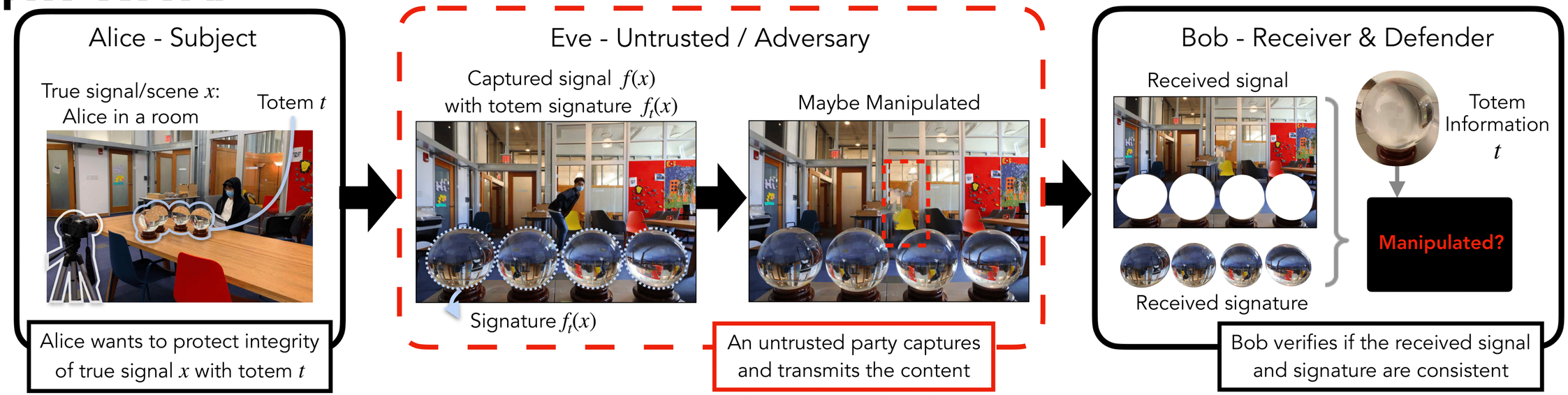
Totems - physical objects for verifying visual integrity

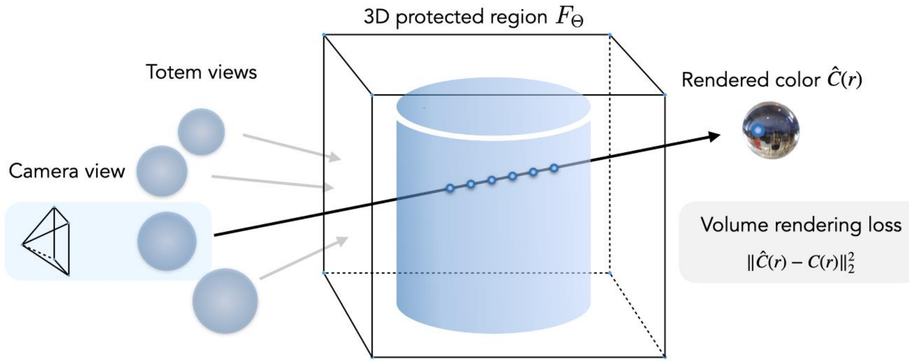


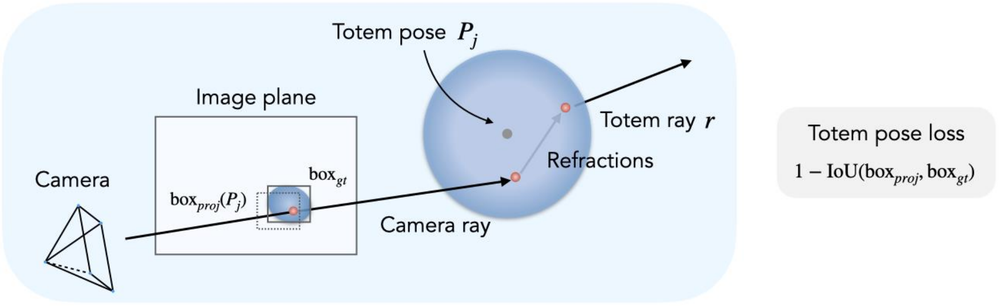
Get a 3D reconstruction of scene by using the warped images in the totem (uses NeRF)

J. Ma et al 2022

Totems - physical objects for verifying visual integrity







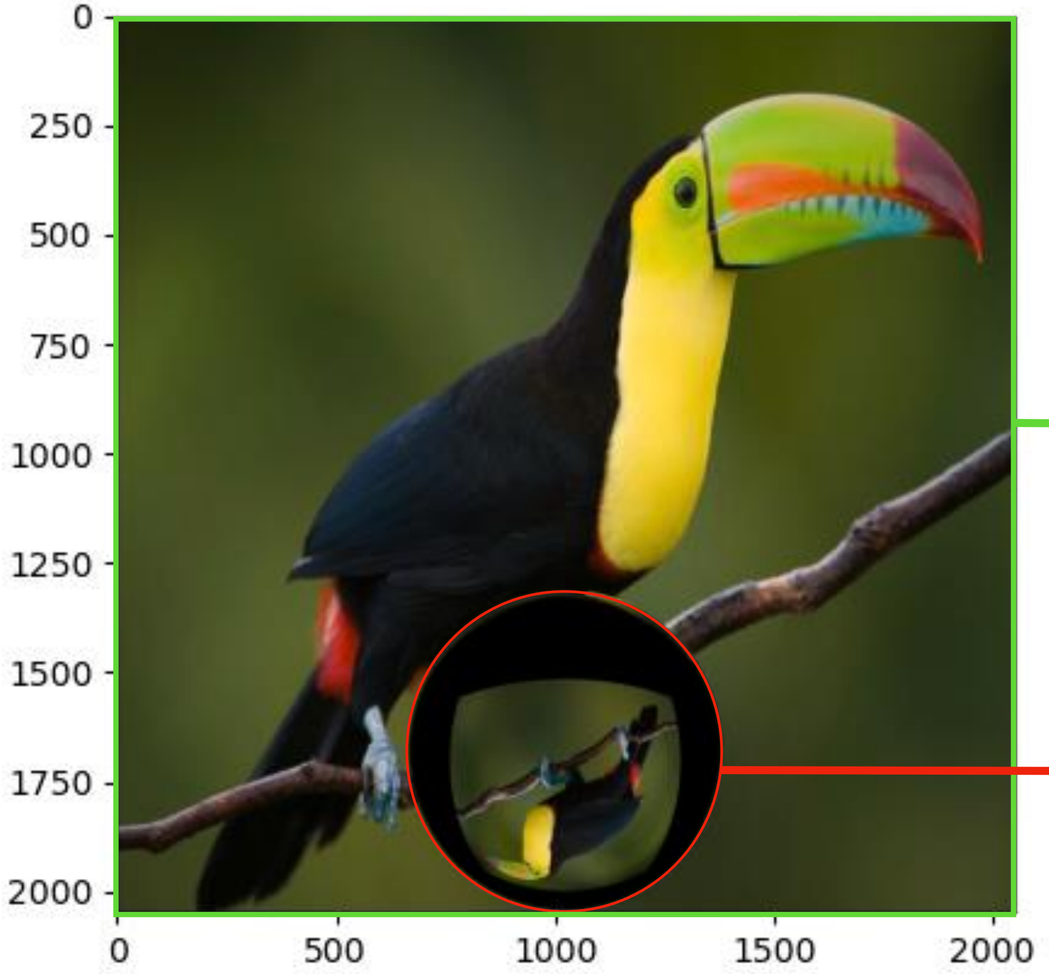
How does the 3D reconstruction work? Light rays are traced from camera to a volumetric region, wherein the neural radiance field is optimized. Along their path, the rays are bent through the totems via analytical computations based on the physical properties (geometry and refractive index) of the material

J. Ma et al 2022

Improvements in our new approach

Totems	Totems++ (ours)
Requires 4 totems in scene	1 totem in scene
Requires refraction model, potentially limiting totem generalization	Learn a model for the refraction
Totem pose ground truth required (unless pose optimization)	Totem pose not required

The totem abstraction



$I(u_o, v_o)$

$I'(u_t, v_t)$

$$I'(u_t, v_t) = \Psi(I(u_o, v_o))$$

$$Err(\vec{I_r}(\vec{I'}; \Psi), \vec{I_m}) = \mathcal{L}(\tilde{\mathbf{\Psi}}^{-1}\vec{I'}, \vec{I_m})$$

Overview

4 stages:

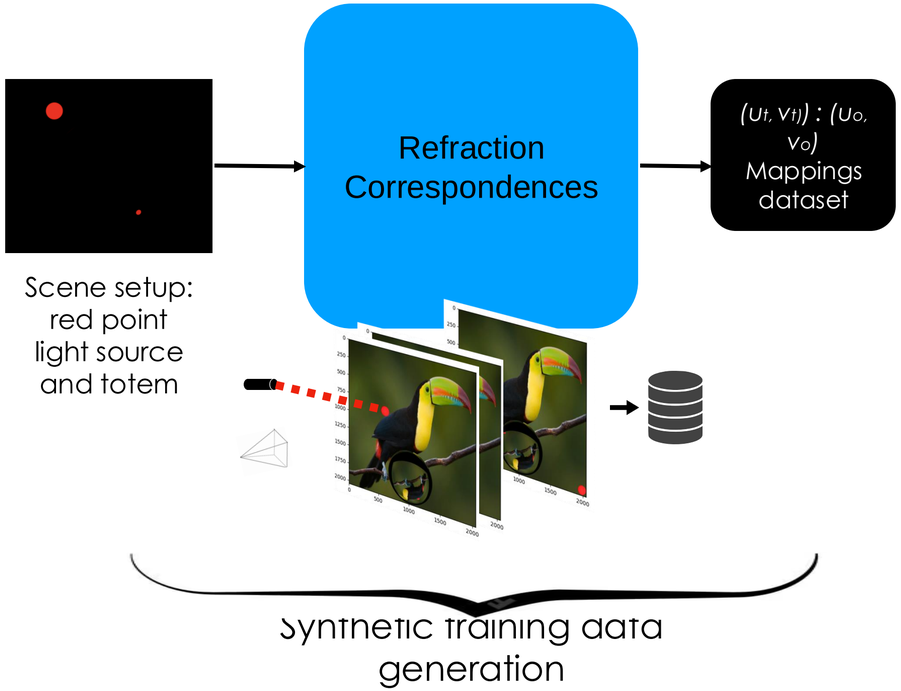
1. Data creation (Mappings Generation): Empirically generate a dataset of scene-totem pixel correspondences

2. NN training: Model the refraction by predicting pixel correspondences

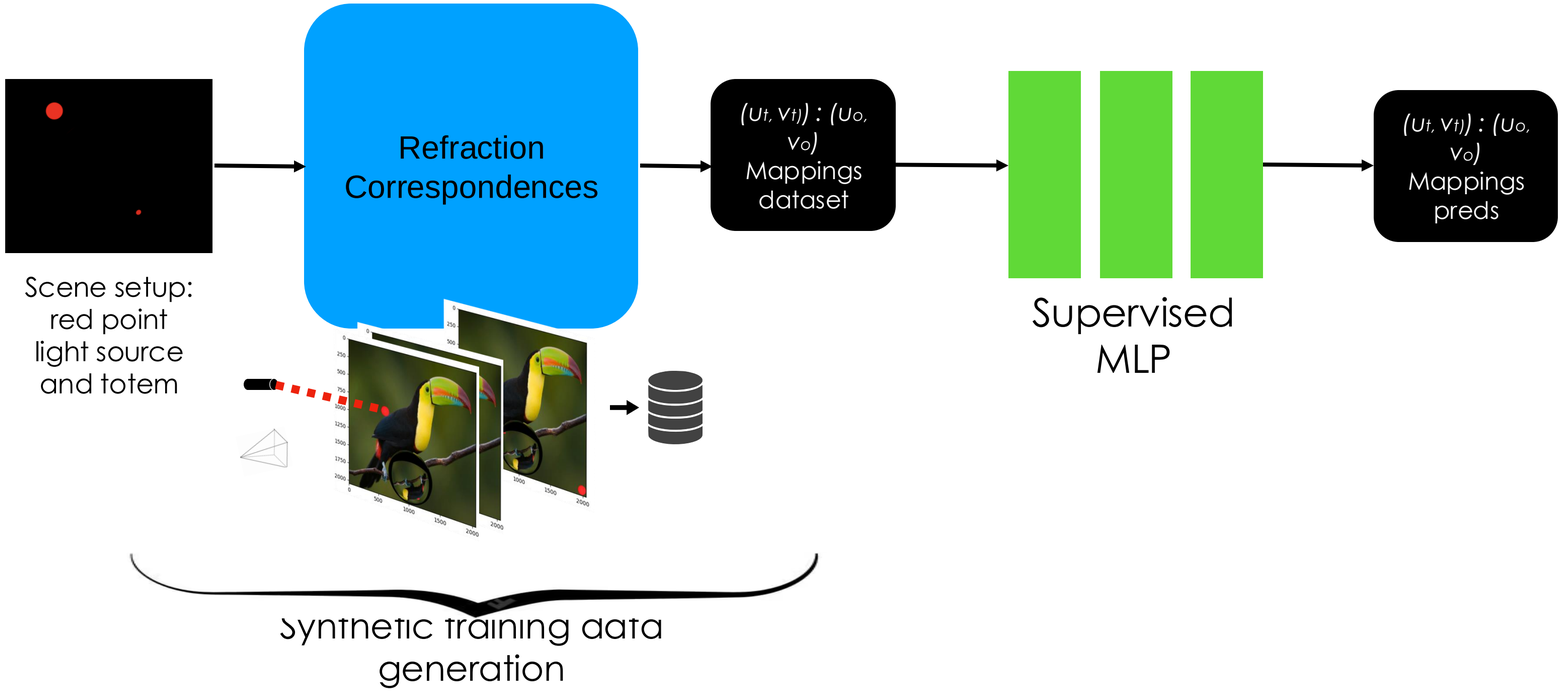
3. Reconstruction: From correspondences, unwarp the totem image back to a lossy approximation the original image

4. Detection: Detect image manipulations using similarity metrics

Overview


Refraction Correspondences
$(u_t, v_t) : (u_o, v_o)$ Mappings dataset
Scene setup: red point light source and totem
Synthetic training data generation

Overview


Refraction Correspondences
$(u_t, v_t) : (u_o, v_o)$ Mappings dataset
$(u_t, v_t) : (u_o, v_o)$ Mappings preds
Supervised MLP
Scene setup: red point light source and totem
Synthetic training data generation

Overview

Lookup table =
{training data U predictions from refraction model}

Scene setup:
red point
light source
and totem

Refraction Correspondences

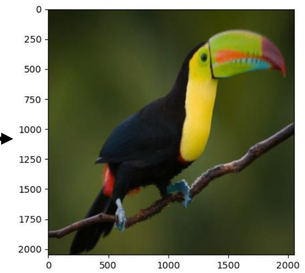
$(u_t, v_t) : (u_o, v_o)$ Mappings dataset

Synthetic training data generation

Supervised MLP

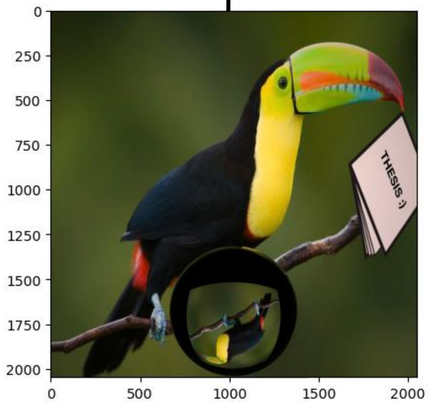
$(u_t, v_t) : (u_o, v_o)$ Mappings preds

Reconstruction



Reconstruction Estimate

Manipulated Image



Overview

Lookup table =
{training data U predictions from refraction model}

Scene setup:
red point
light source
and totem

Refraction Correspondences

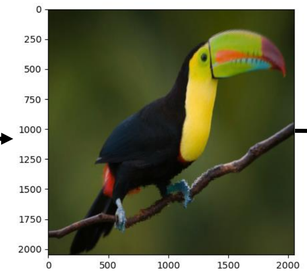
$(U_t, V_t) : (U_o, V_o)$ Mappings dataset

Synthetic training data generation

Supervised MLP

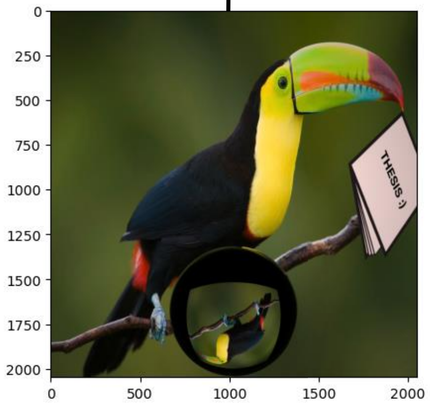
$(U_t, V_t) : (U_o, V_o)$ Mappings preds

Reconstruction

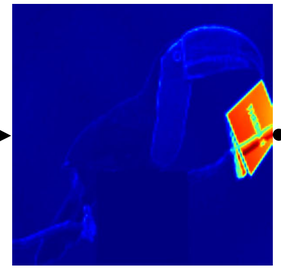


Reconstruction Estimate

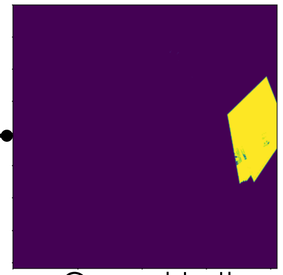
Manipulated Image



Detection



Inconsistency Heatmap

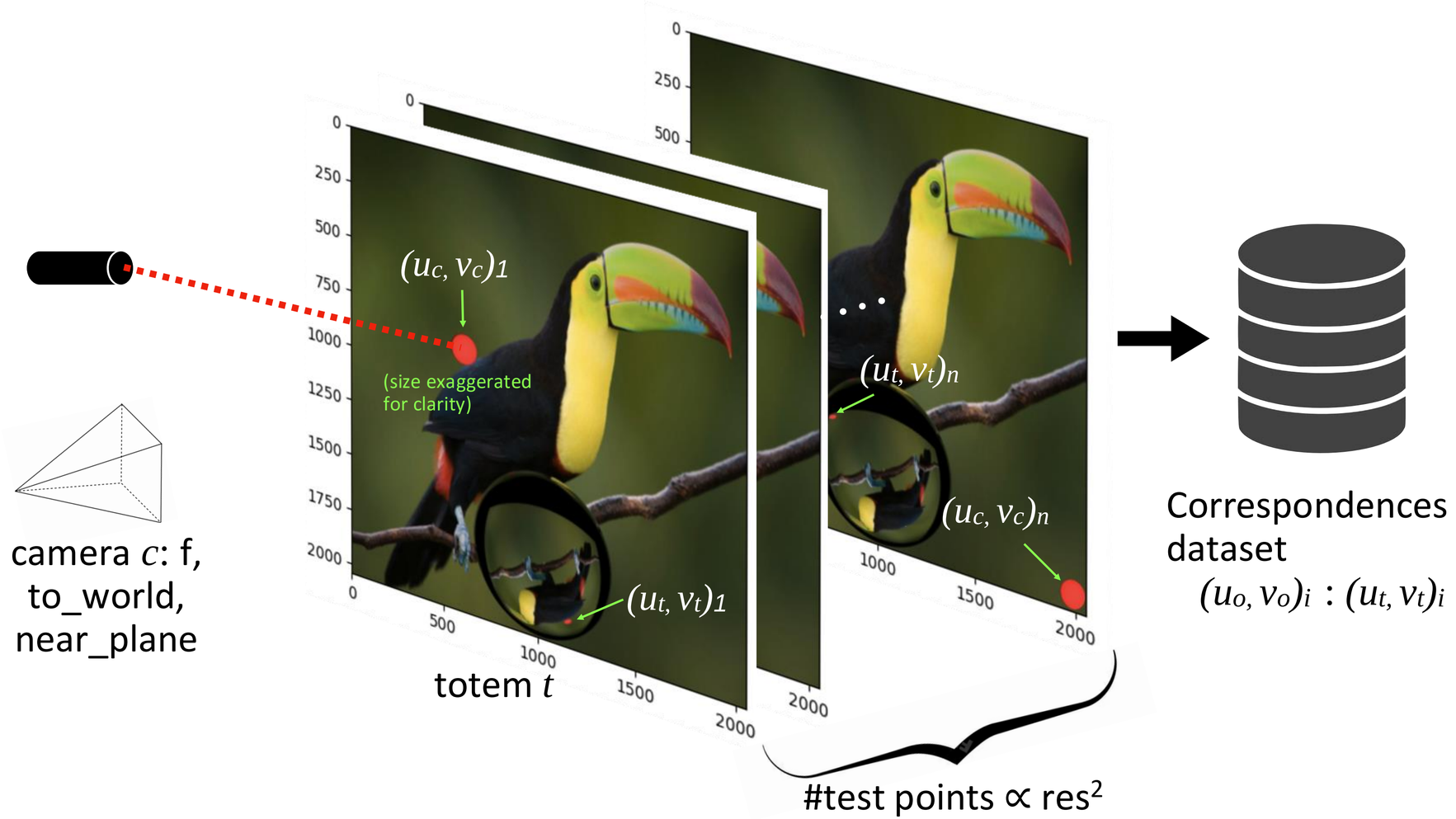


Ground truth manipulation mask (only for evaluation purposes)

(accuracy = 0.99)

The computational bottleneck: Mappings Generation

A 2D, geometry-independent approach



- Computational bottleneck
- Heuristic-based optimizations, GPU parallelization, and finally using a NN to interpolate sparse training data make this process at least as fast as prior work's bottleneck

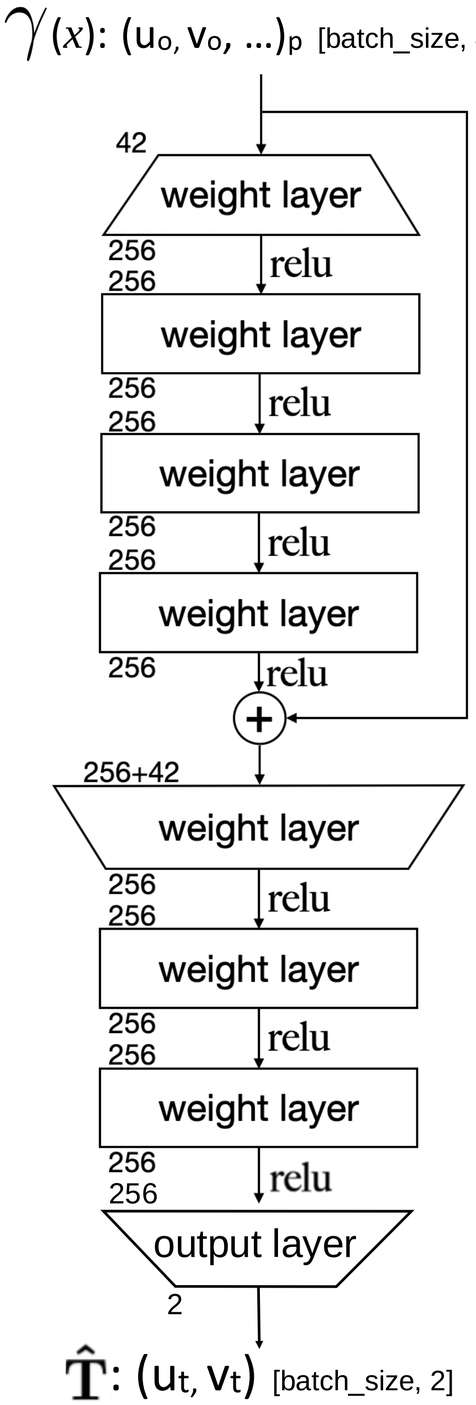
Learning the refraction model

Neural net training

- Goal: build a model to predict the refraction correspondences between scene and totem pixel
  - Input: (u_o, v_o) Output: (u_t, v_t)
- For a sphere (and our dataset) and finite resolution, such a function is a smooth many-to-one function
- Use an 8 layer MLP similar to the NeRF architecture
- Let $\hat{\mathbf{T}}$ be the [n, 2] totem pixel estimate and T be the ground truth totem pixel. Then optimize according to:

$$\mathrm{MSE}(\hat{\mathbf{T}}, \mathbf{T}) = \frac{1}{B \times 2} \sum_{i=1}^{B} \sum_{j=1}^{2} (\hat{\mathbf{T}}^{\mathbf{ij}} - \mathbf{T}_{\mathrm{gt}}^{\mathbf{ij}})^2,$$

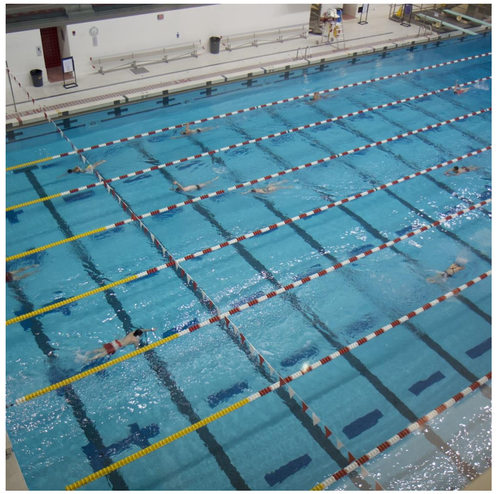
where B is the batch size.

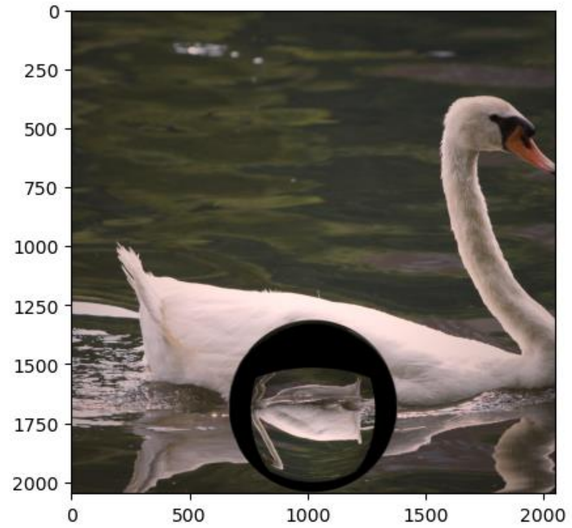
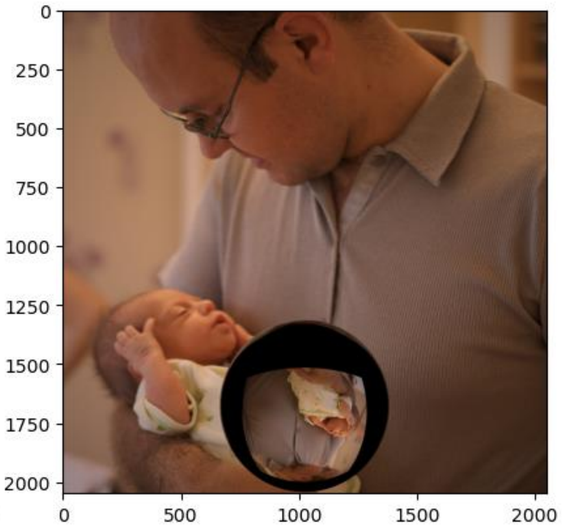
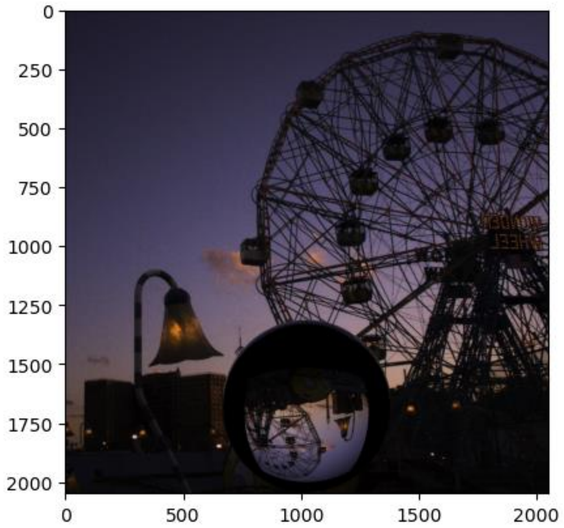
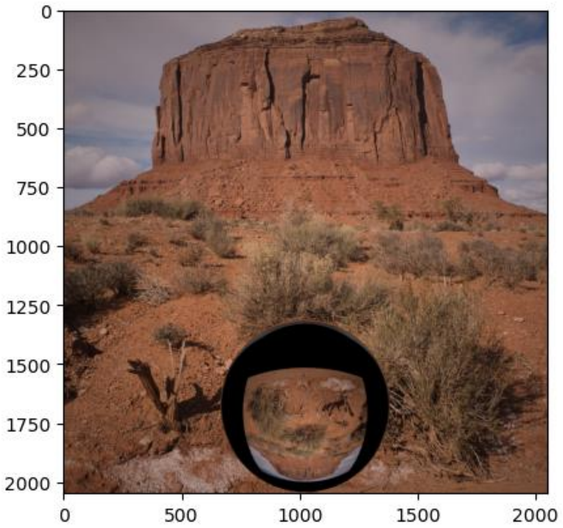


Reconstruction

Dataset

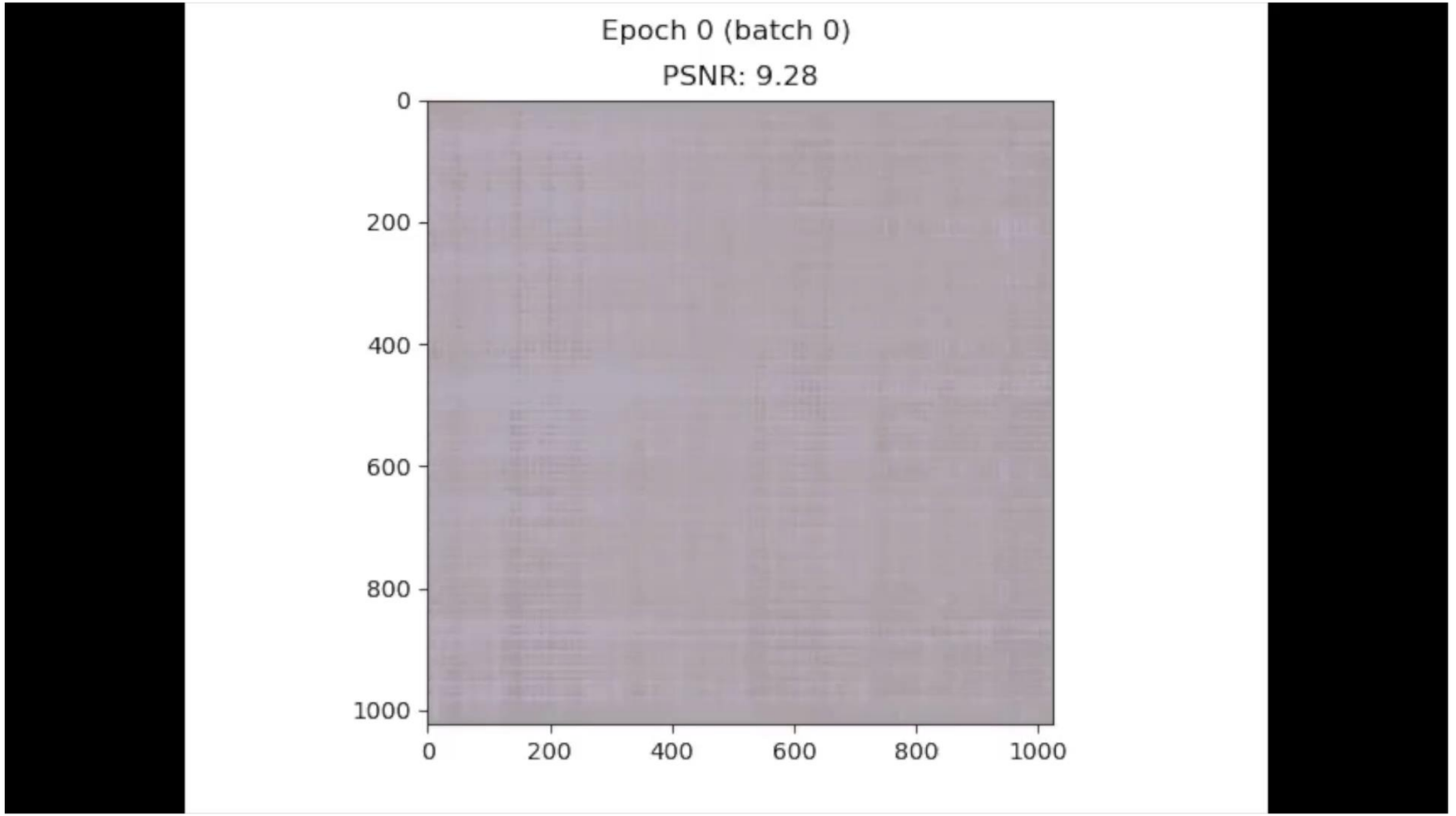



...


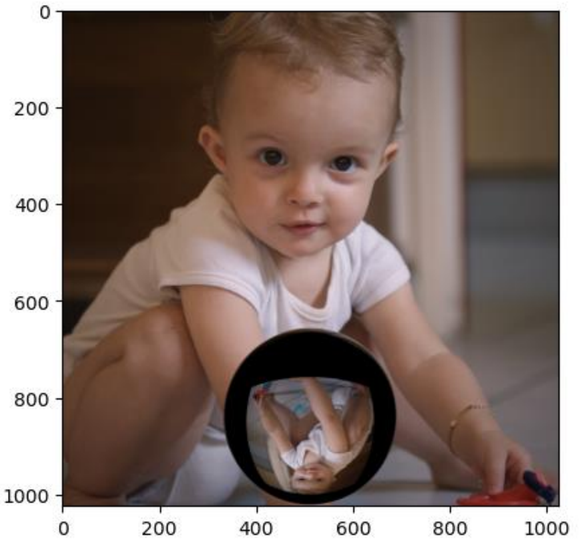



...


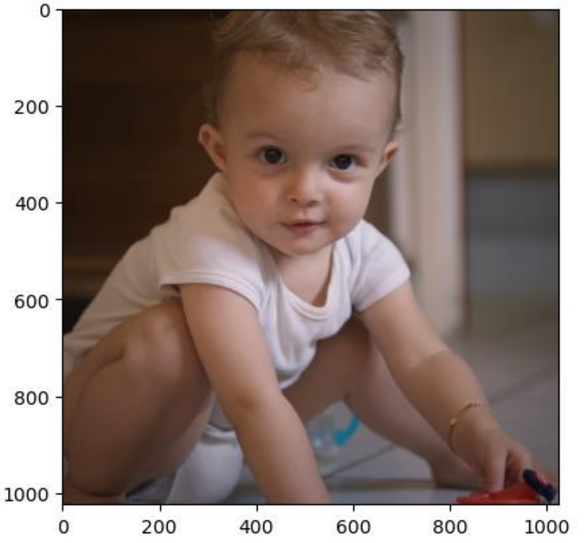
MIT-Adobe FiveK Dataset

Unwarping progress during different steps of NN training

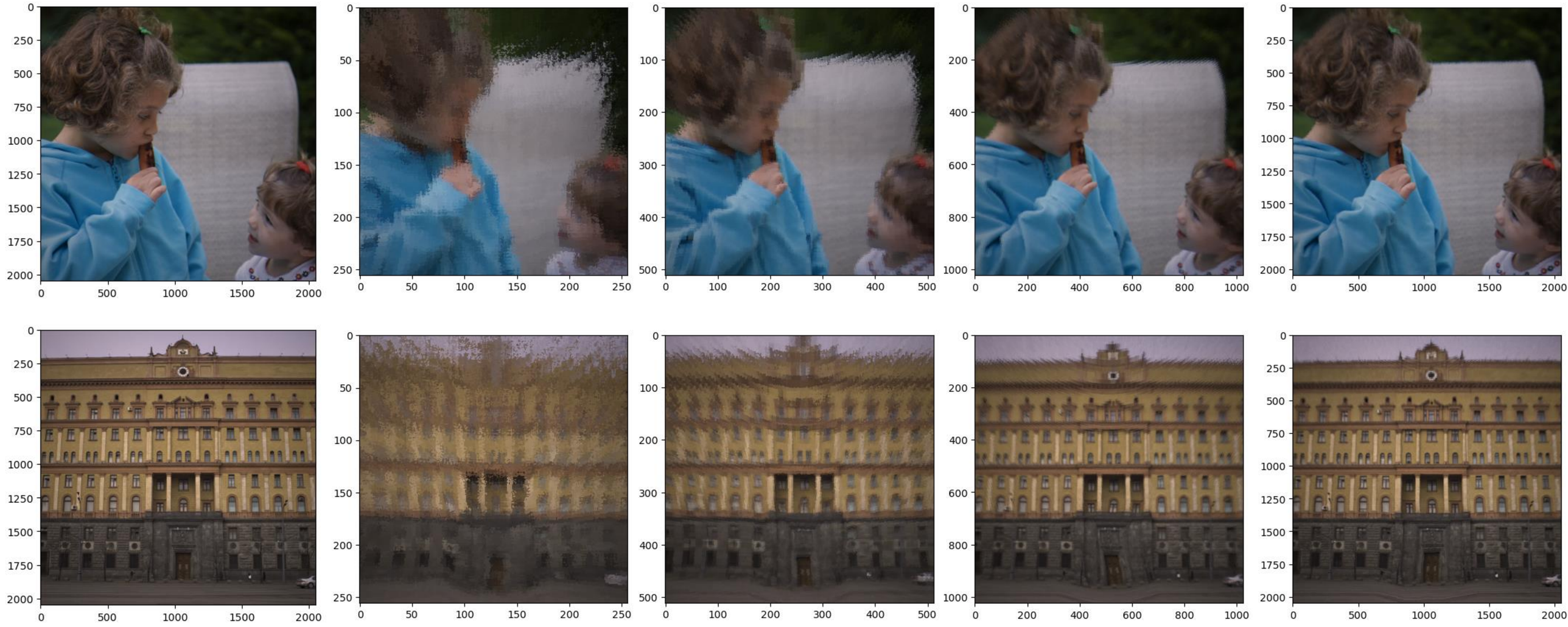


Ground truth reference





Trying higher resolutions



Original 256 512 1024 2048

Detection

Detection on manipulated images

Toy example: colorpatch

$$Err(\vec{I_r}(\vec{I'};\Psi),\vec{I_m}) = \mathcal{L}(\tilde{\mathbf{\Psi}}^{-1}\vec{I'},\vec{I_m})$$

1. Generate manipulated dataset with randomly placed and randomly sized color patches on the image

2. Segment the image according to patch size p, and obtain the L1 patch-wise loss between reconstruction $\vec{I_r}$ and manipulated image $\vec{I_m}$

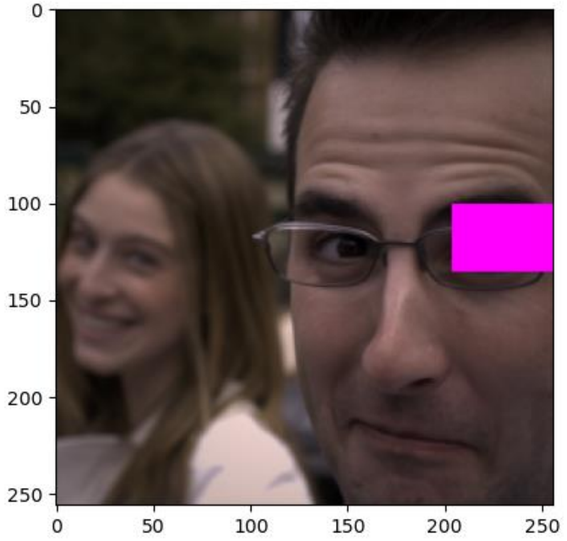
3. To evaluate detection performance for a single datapoint, use the average precision (AP) score, or the mean AP (mAP) over the entire dataset:

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n$$

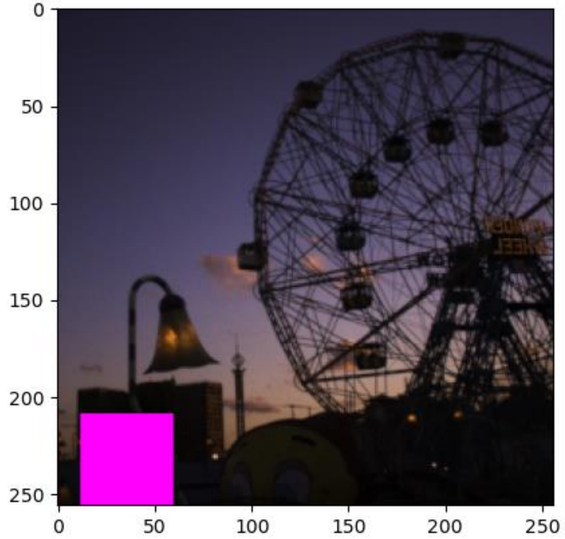
$$mAP = \frac{1}{N} \sum_{n=1}^{N} \sum_j (R_{j,n} - R_{j-1,n}) P_{j,n},$$

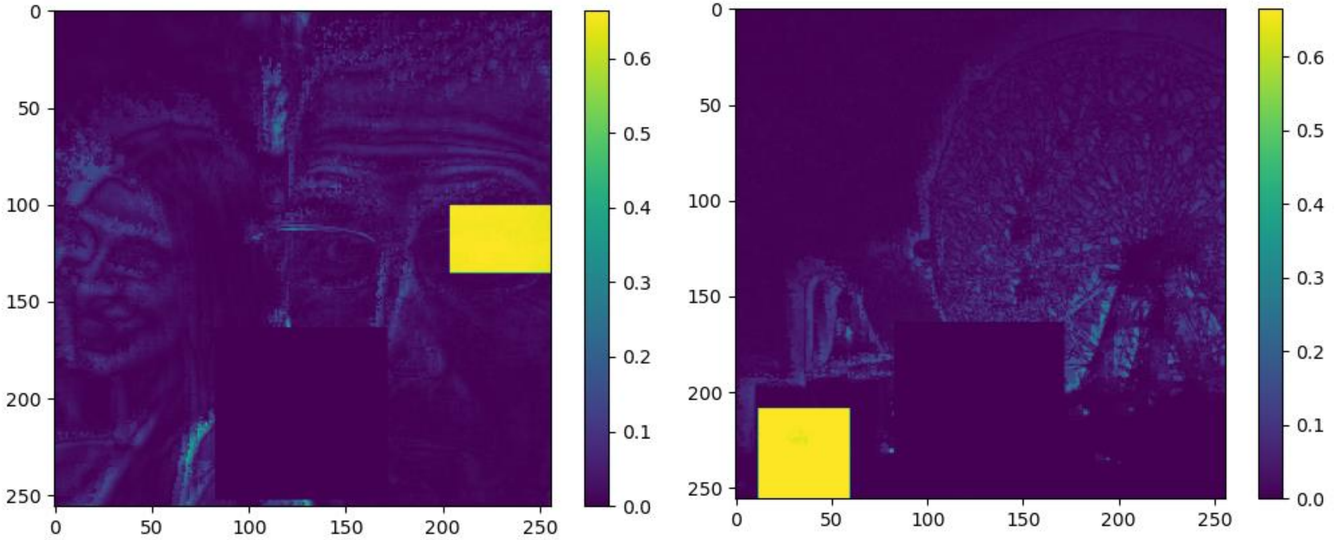
where $R_{j,n}$ and $P_{j,n}$ are the recall and precision (respectively) of the n^{th} image's grid of classifications and j^{th} threshold.

AP: 1.0



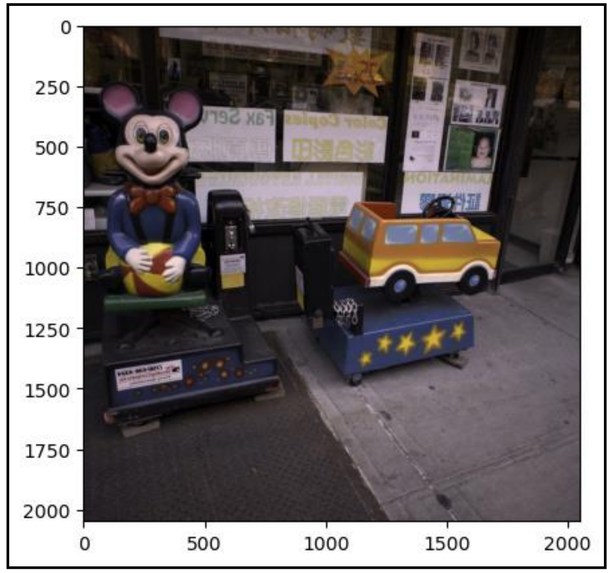
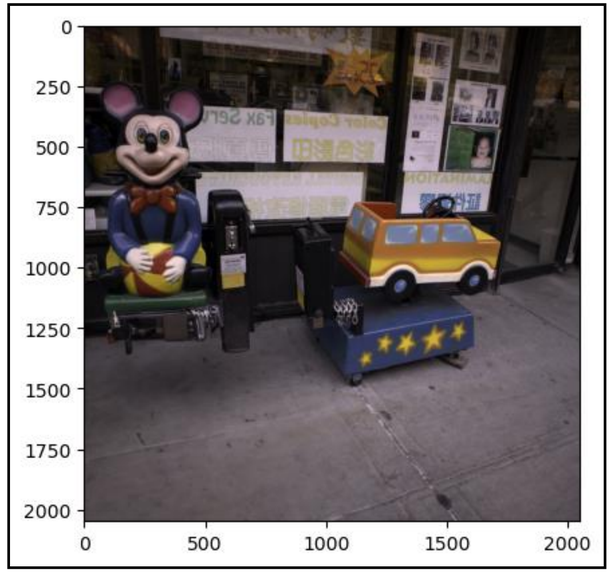
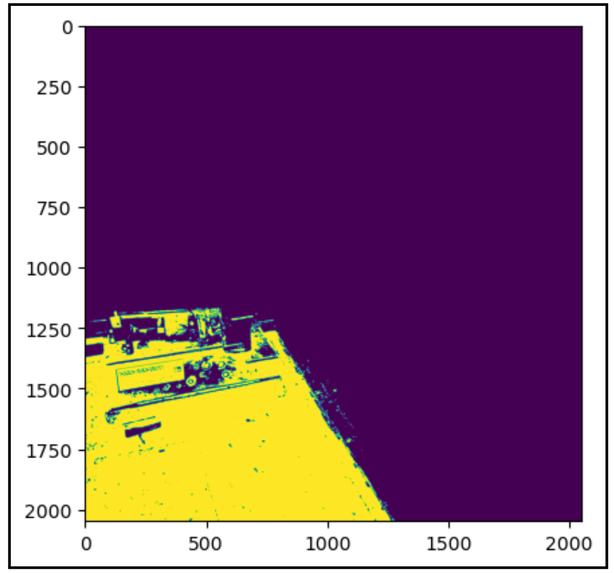
AP: 1.0





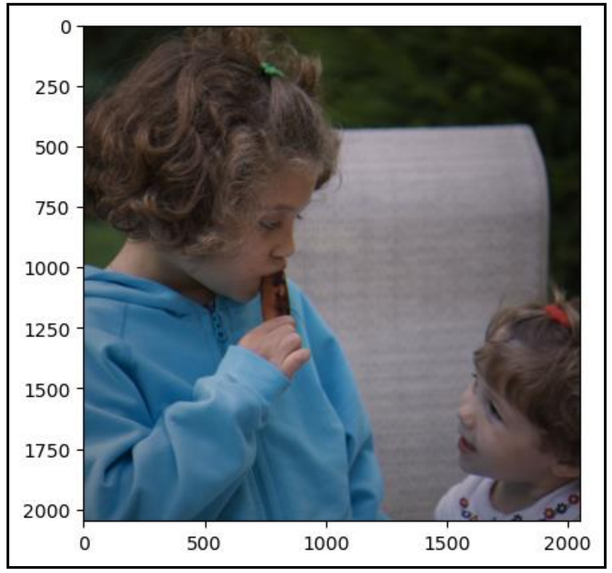
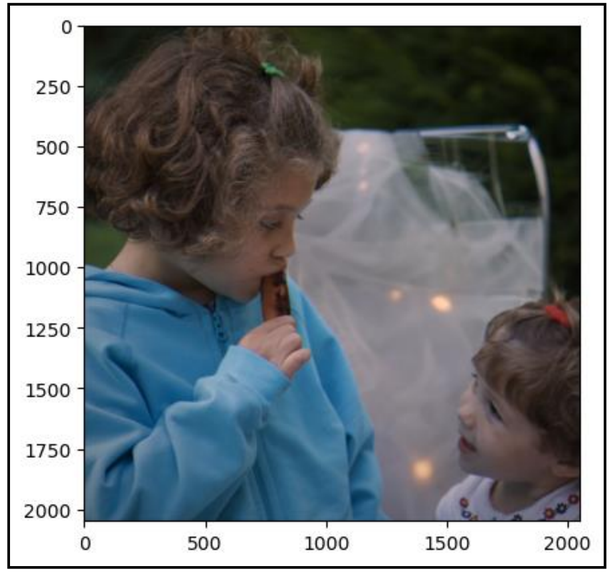
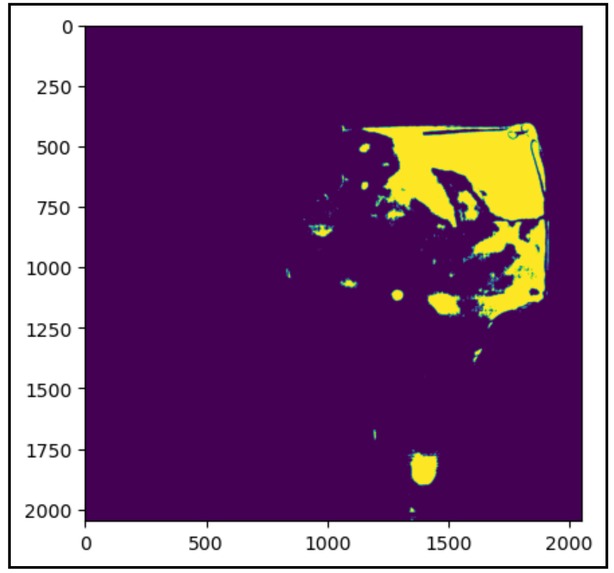
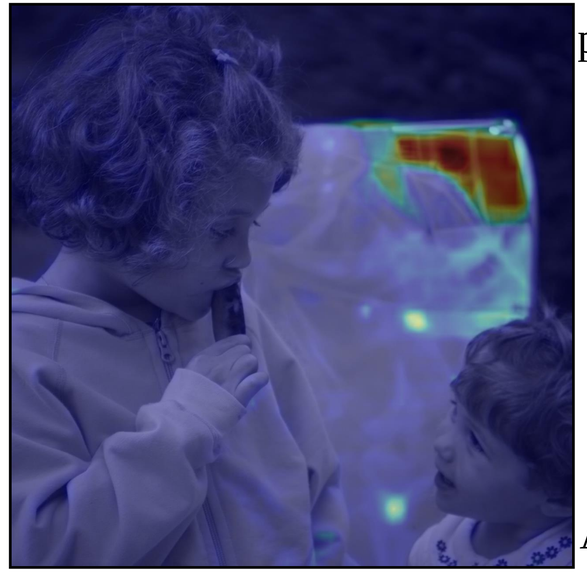
Top: manipulated image; bottom: inconsistency heatmap

Detection results (textural manipulations)


p32

AP 0.96

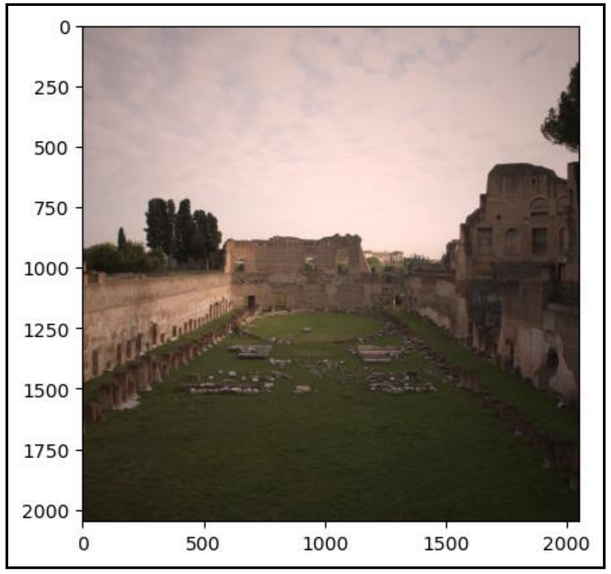
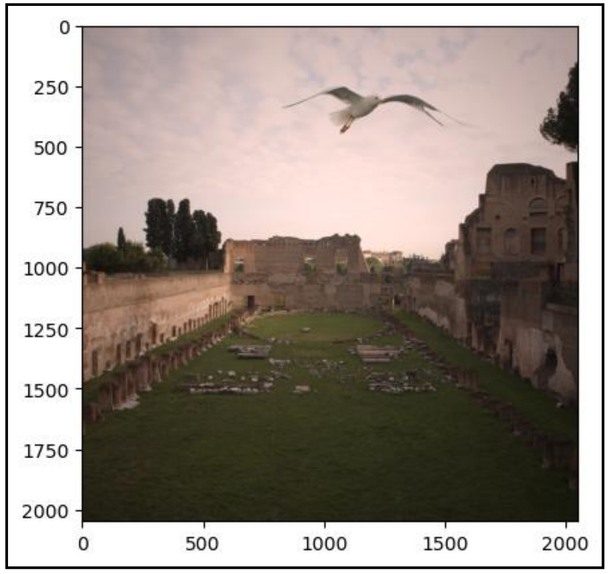
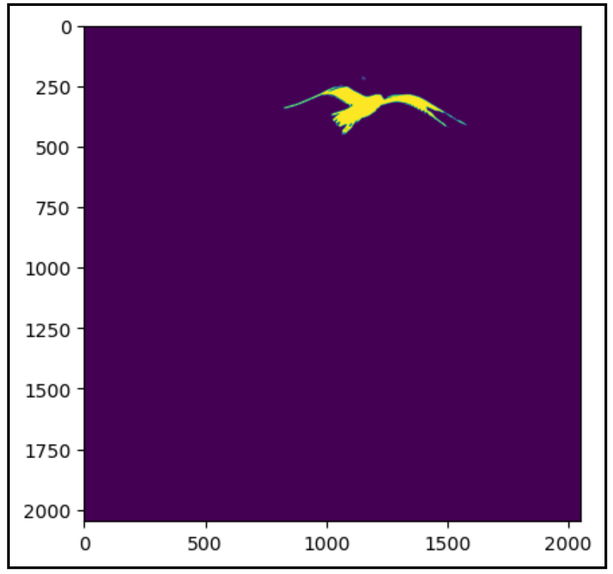
p32

AP 0.93

Original | Manipulated | Manipulation mask | Inconsistency Heatmap

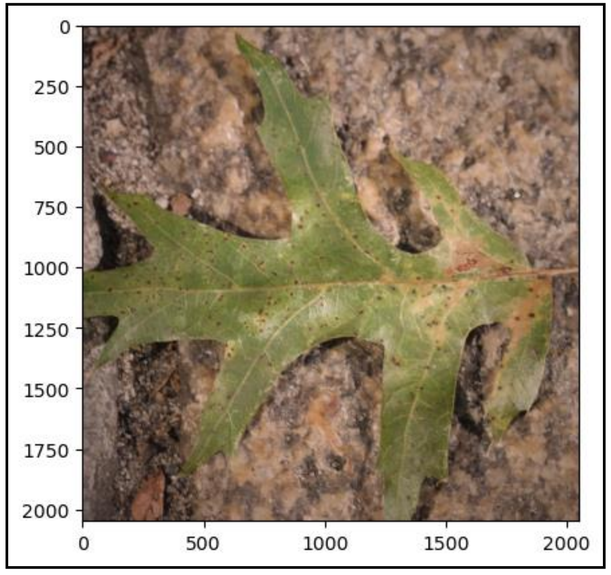
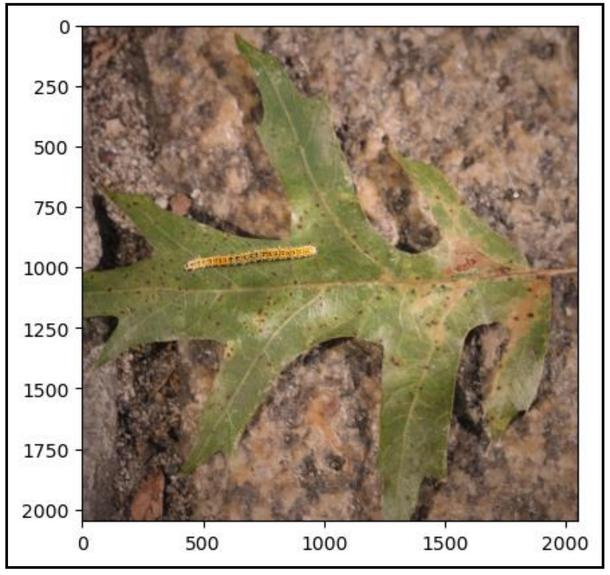
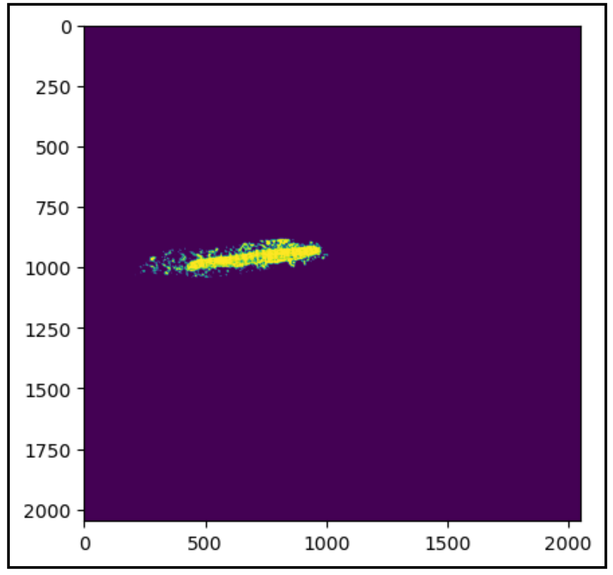
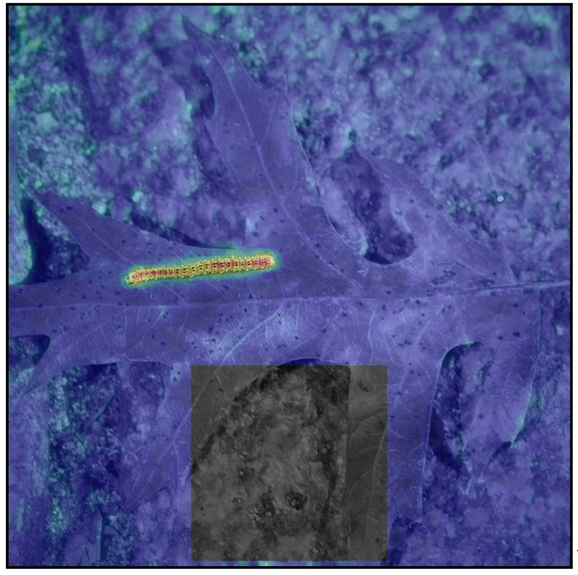
Detections are made over a designated patch size and loss metric and evaluated using average precision score

Detection results (image splices)


p32

AP 0.86

p32

AP 0.78

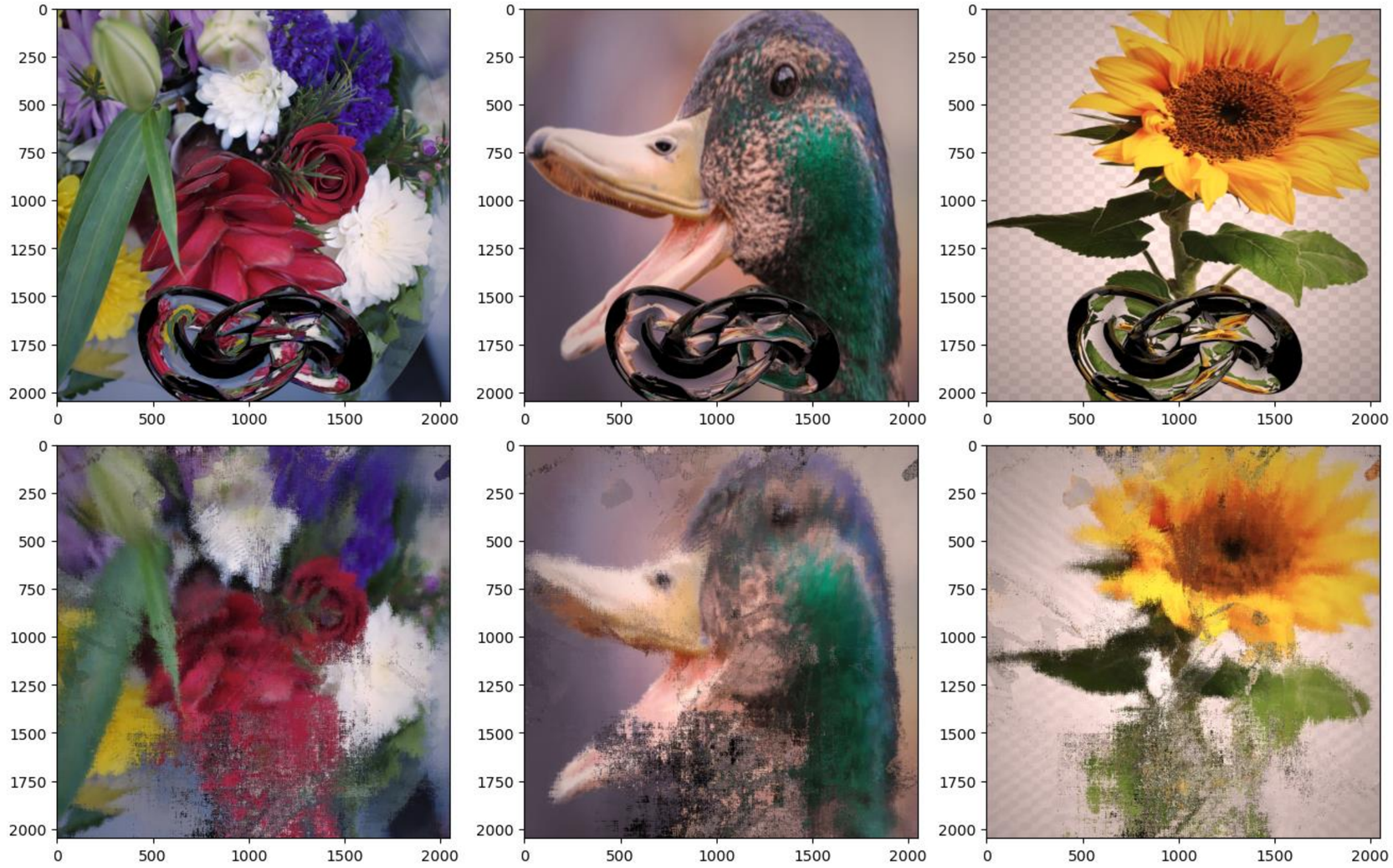
Original | Manipulated | Manipulation mask | Inconsistency Heatmap

Can we generalize to other shapes?

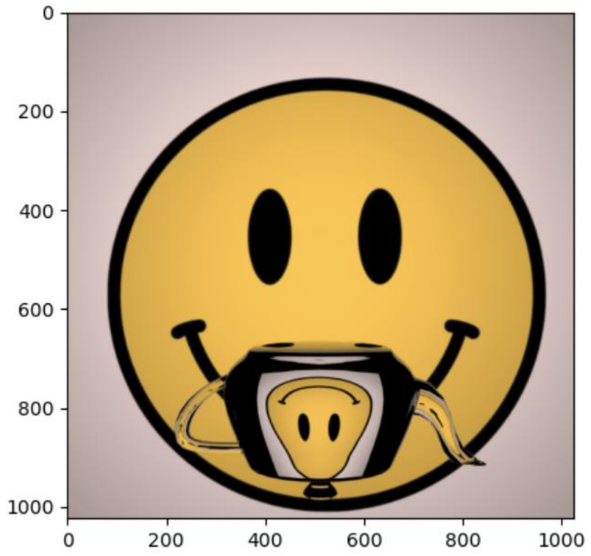
- By enabling independence of a closed form of the refraction, we open the door to potentially more secure geometries
- Complicating factors include multiple refractions and high frequency mapping function
- With algorithmic changes and addition of positional encodings in the model, can achieve high detection accuracy on very complex shapes

Original Images

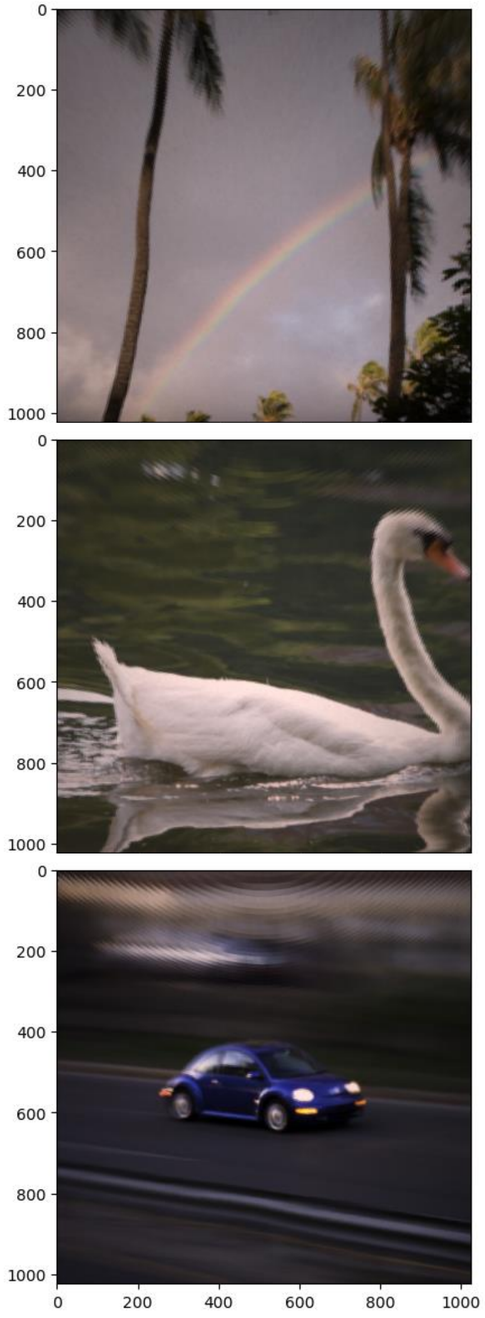
Reconstructions



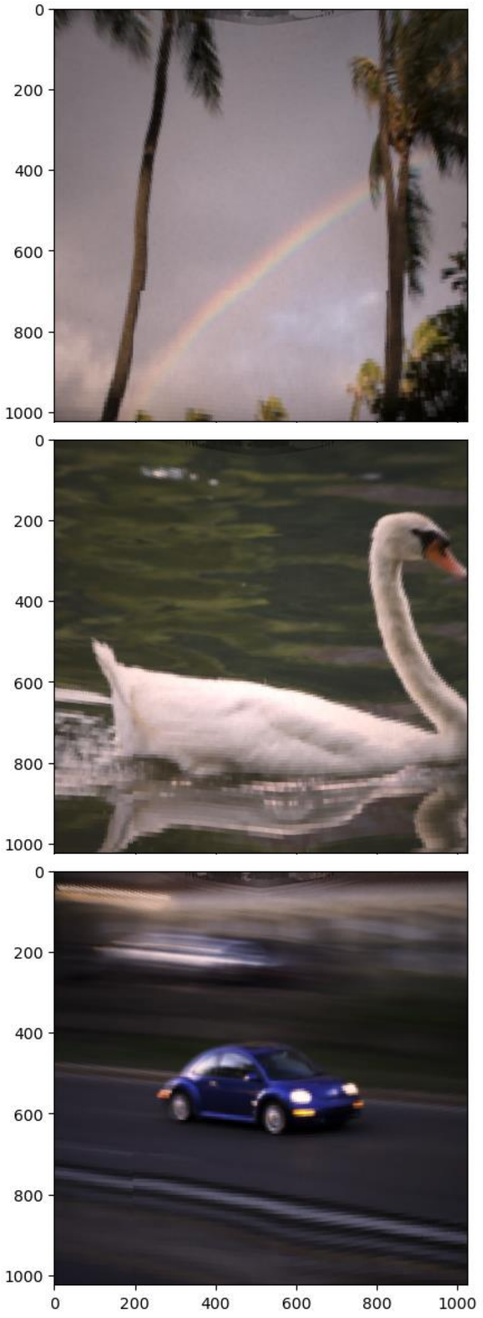
Some other results



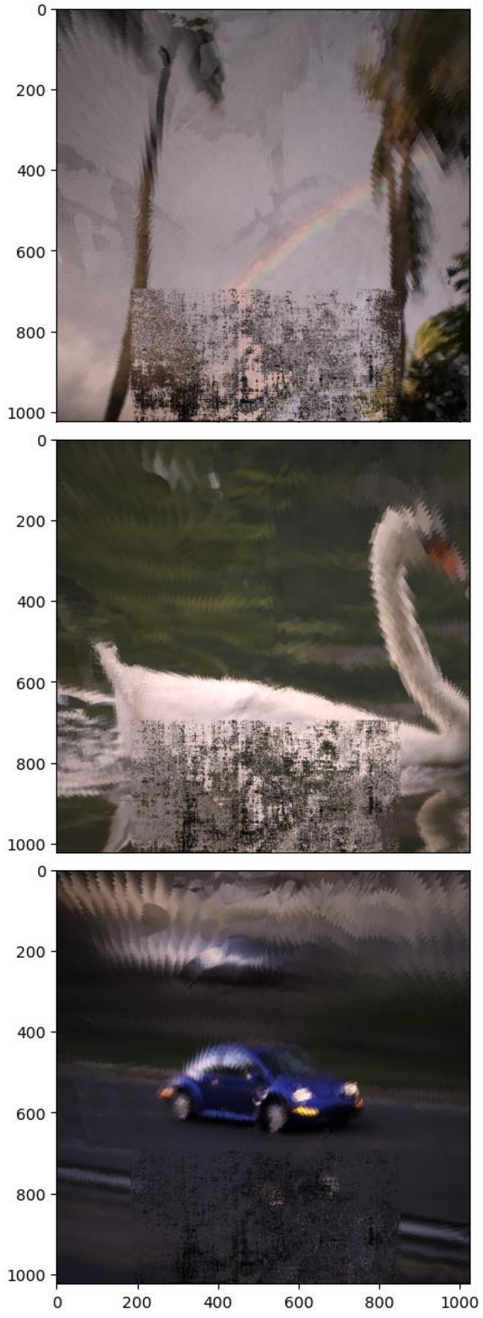
Teapot for reference



Sphere$_{1024}$



Teapot$_{1024}$



Knot$_{1024}$

Initial Detection Results

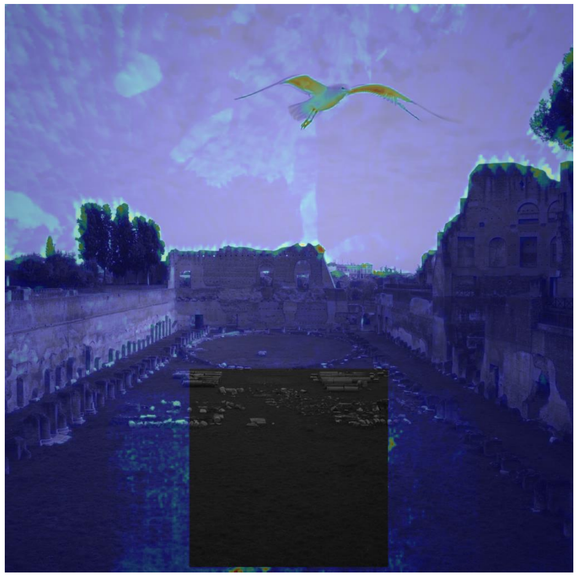
Teapot

AP 0.94

0.61

Knot

AP 0.88



0.59

Questions?

Backup

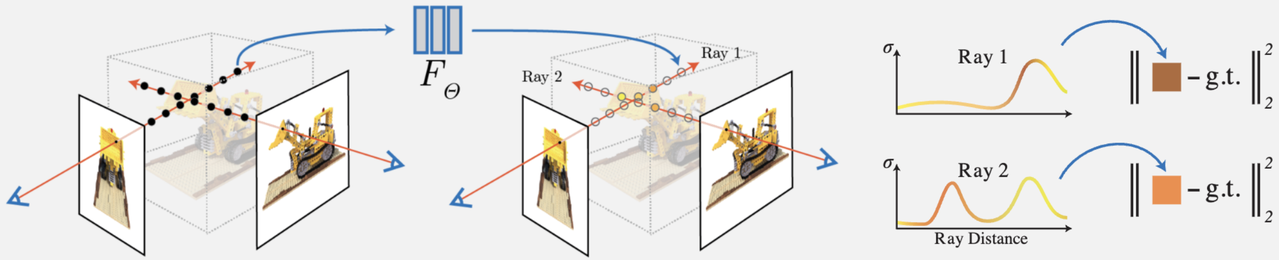
NeRF

Abstract & Method

We present a method that achieves state-of-the-art results for synthesizing novel views of complex scenes by optimizing an underlying continuous volumetric scene function using a sparse set of input views.

$$(x,y,z,\theta,\phi) \rightarrow F_\Theta \rightarrow (RGB\sigma)$$

Our algorithm represents a scene using a fully-connected (non-convolutional) deep network, whose input is a single continuous 5D coordinate (spatial location (x, y, z) and viewing direction (θ, φ)) and whose output is the volume density and view-dependent emitted radiance at that spatial location.



We synthesize views by querying 5D coordinates along camera rays and use classic volume rendering techniques to project the output colors and densities into an image. Because volume rendering is naturally differentiable, the only input required to optimize our representation is a set of images with known camera poses. We describe how to effectively optimize neural radiance fields to render photorealistic novel views of scenes with complicated geometry and appearance, and demonstrate results that outperform prior work on neural rendering and view synthesis.

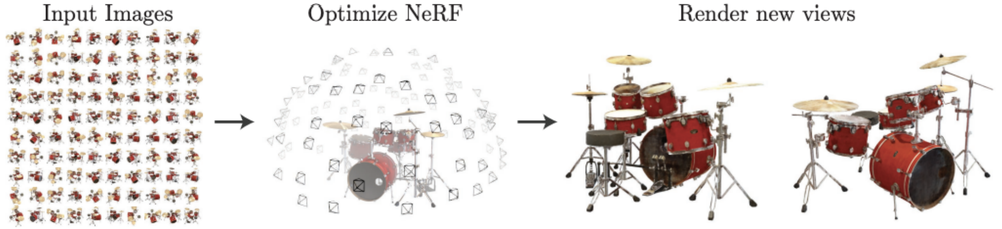


Fig. 1: We present a method that optimizes a continuous 5D neural radiance field representation (volume density and view-dependent color at any continuous location) of a scene from a set of input images. We use techniques from volume rendering to accumulate samples of this scene representation along rays to render the scene from any viewpoint. Here, we visualize the set of 100 input views of the synthetic *Drums* scene randomly captured on a surrounding hemisphere, and we show two novel views rendered from our optimized NeRF representation.

M. Tancik et al 2020

Mappings generation

Method at a high level

Scene setup: point light source (red laser dot) at depth d, totem

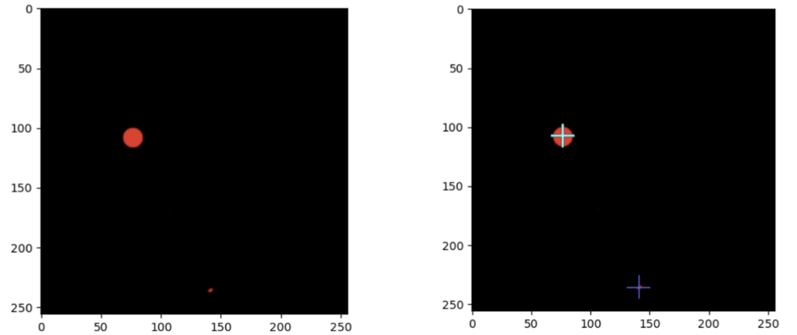
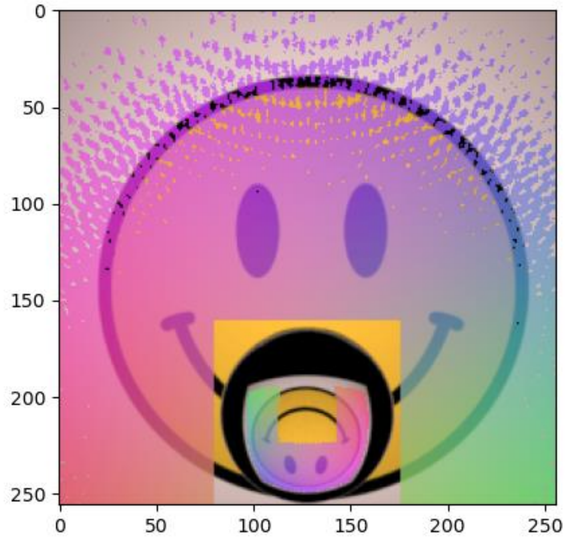


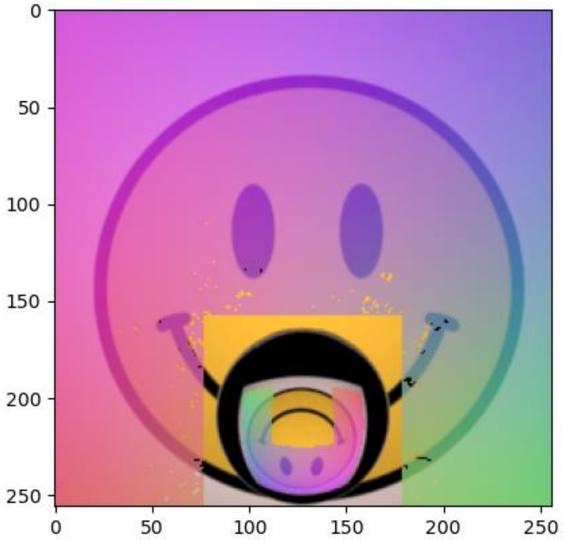
Figure 1-2: Example of finding the location of the two pixels for the correspondence (exaggerated red dot size) on a resolution 256 rendering. Crosshairs show the calculated (u_c, v_c) (teal) at $(77, 108)$ and detected refraction (u_t, v_t) (purple) at $(141, 236)$.

For every test point world coord (u, v, d) in the sweep over X and Y:

1. Analytically convert laser coordinates to image coordinates (u_c, v_c) as seen by the camera *c*
2. Render the area within the bounding box surrounding the totem
3. Using color thresholding, look for a clusters of red pixels and use the centroid of the cluster as the refraction (u_t, v_t)
4. Store the pair (u_c, v_c) : (u_t, v_t) in the dataset

Initial results





Parameters needing tuning:

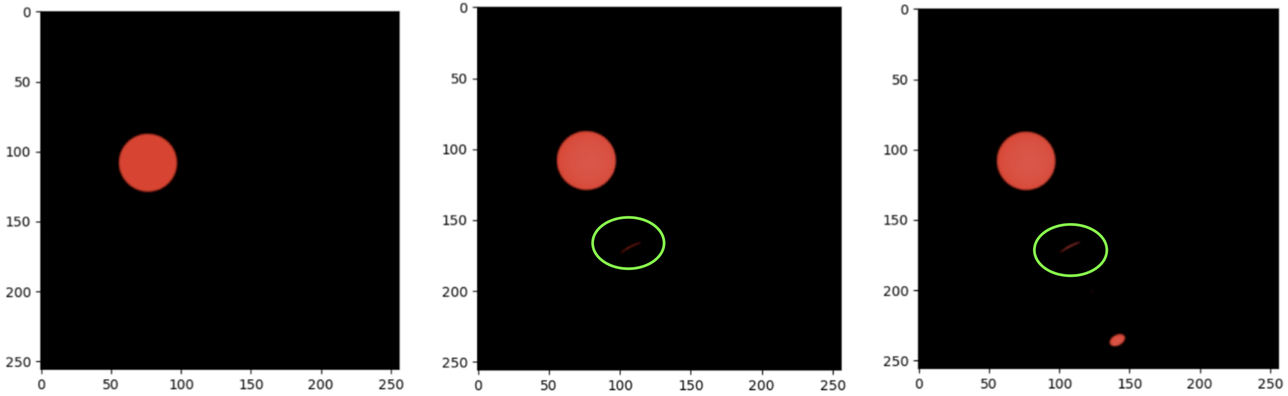
- Laser dot size: must be large enough to be captured in most compressed regions of the scene, but small enough for precise mapping in the totem

- Threshold: if too low picks up rendering artifacts and quantization errors that cause non localized clusters of red pixels in the totem, if too high can lose the actual refraction

Other issues with sweep implementation

Rendering artifacts

- Undesired reflections & higher order effects - usually 1-2 other clusters of pixels picked up by thresholder



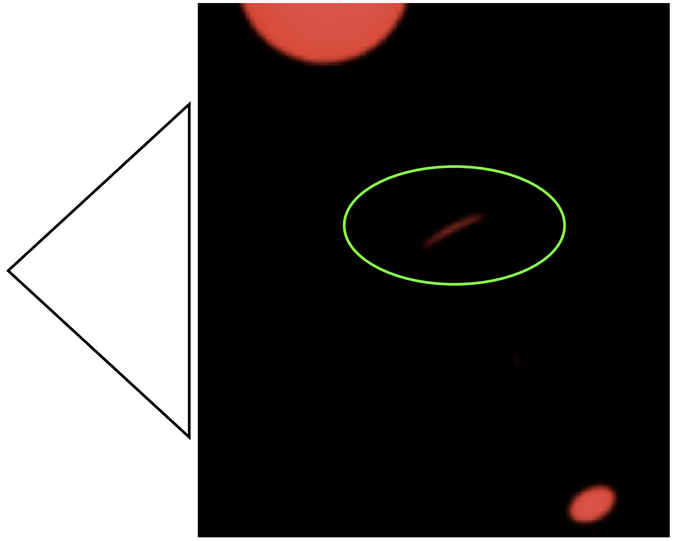
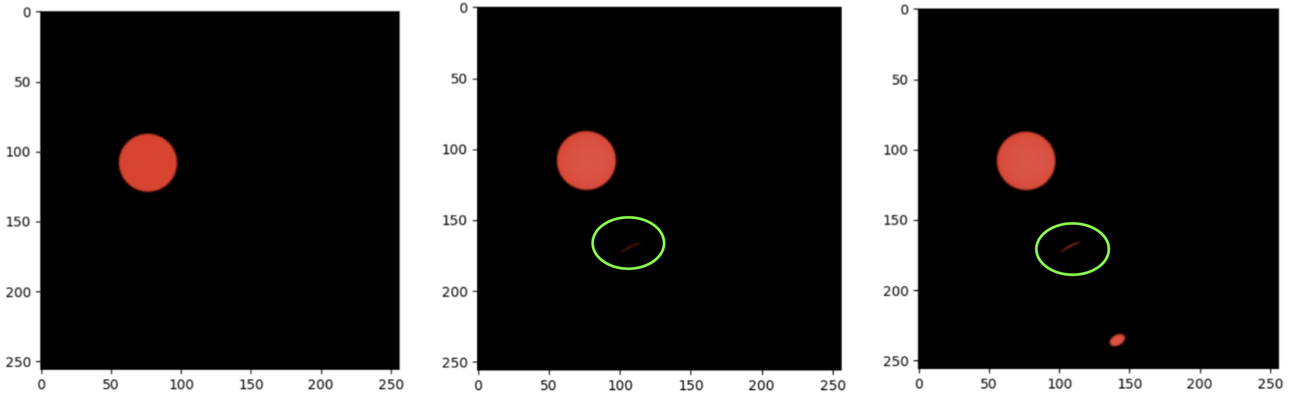


Figure 1-6: Renderings of the red dot (exaggerated for clarity) and the result in the totem for different values of `max_depth`. From L to R: Max depth set to 1 bounce. Only the red dot is rendered. Max depth set to 2 bounces; both the red dot and the reflection off the edge of the totem are visible. Max depth set to 3 bounces; the refraction is now rendered as well.

Other issues with sweep implementation

Rendering artifacts

- Undesired reflections & higher order effects - usually 1-2 other clusters of pixels picked up by thresholder



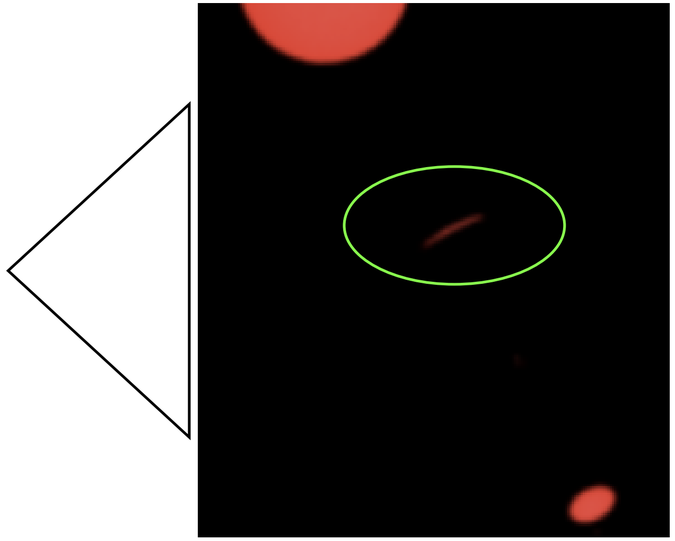


Figure 1-6: Renderings of the red dot (exaggerated for clarity) and the result in the totem for different values of `max_depth`. From L to R: Max depth set to 1 bounce. Only the red dot is rendered. Max depth set to 2 bounces; both the red dot and the reflection off the edge of the totem are visible. Max depth set to 3 bounces; the refraction is now rendered as well.

- Solution: discard up to 2 of the dimmer clusters

Mappings generation

Method at a high level (augmented)

Scene setup: point light source (red laser dot) of scale *s* at depth d, totem

For every test point world coord (u, v, d) in the sweep over U and V:

1. Analytically convert laser coordinates to image coordinates (u_c, v_c) as seen by the camera *c*

2. Render the area within the bounding box surrounding the totem

3. Using color thresholding with threshold *t*, look for a clusters of red pixels and use the centroid of the cluster as the refraction (u_t, v_t)

    1. Avoid broken clusters by discarding clusters that are within distance d of each other, keeping only one cluster

    2. If >1 remaining cluster after this, discard dimmer clusters

4. Store the pair (u_c, v_c) : (u_t, v_t) in the dataset

Mappings Generation

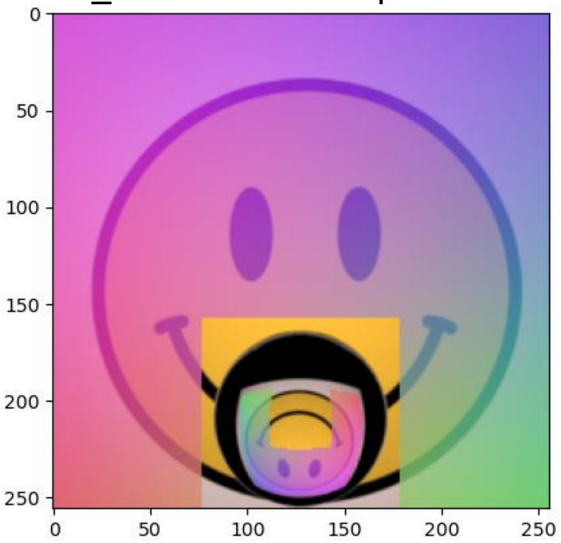
With a combination of tuning the dot size, threshold, and using cluster grouping + filtering, achieve the following results:

Correspondence dataset sizes at different resolutions

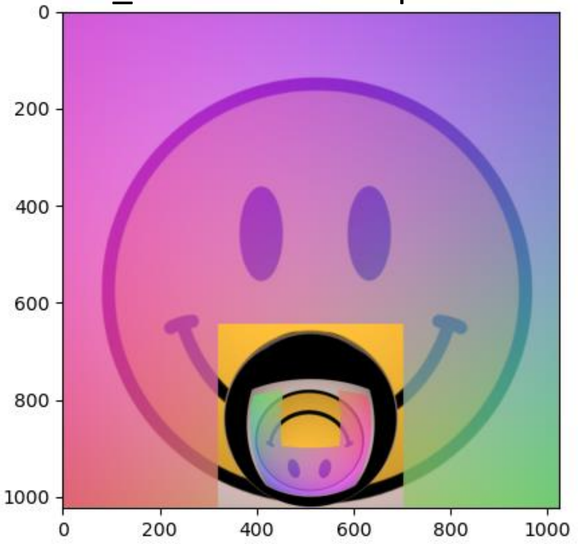
Resolution	256	512	1024	2048
Number of unique (u_t, v_t)'s found in sweep (effective totem resolution)	2,122	8,227	33,625	133,265
Number of (u_c, v_c) datapoints in sweep with refraction found	55,540	222,160	900,088	3,593,708
Num pixels tested (= res*res - excluded region)	55,540	222,160	900,088	3,593,708
Fraction correspondences found	1.0	1.0	1.0	1.0
Percent unique correspondences (100 * row 1 / row 2)	3.82%	3.70%	3.74%	3.71%

More camera pixels than totem pixels
=> **many-to-one mappings**

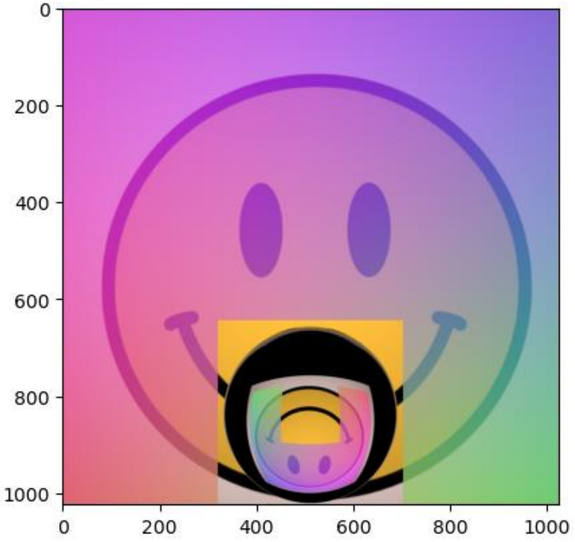
r_256 color correspondence



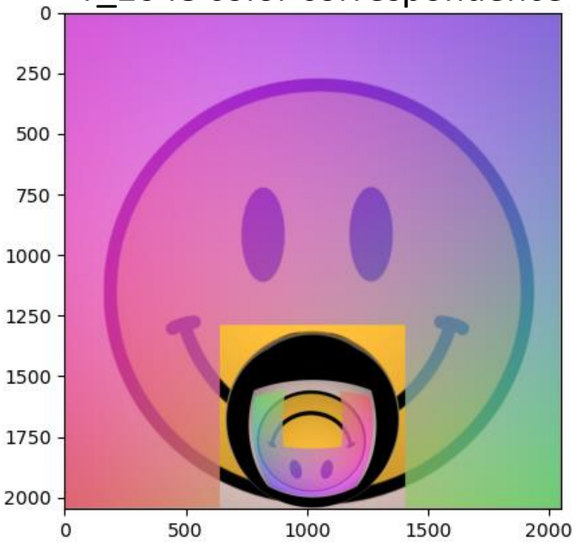
r_512 color correspondence



r_1024 color correspondence



r_2048 color correspondence



Computational Efficiency

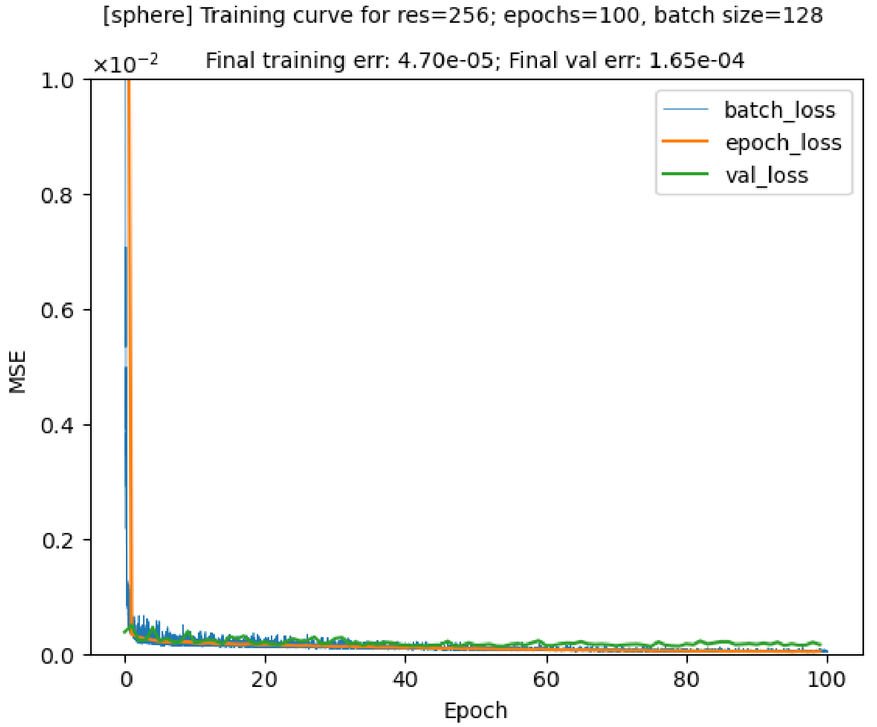
With Python parallelization and using CUDA optimizations for the rendering backend, our initial approach comprising a full density sweep runs in approximately 5 minutes for resolution 256, 25 minutes for resolution 512, 5 hours for resolution 1024, and 30 hours for resolution 2048, when distributed across 16 NVIDIA Tesla V100-SXM2-32GB GPUs. The polynomial increase in runtime is due to quadratic growth in number of frames to process with resolution, a polynomial increase in per-frame processing (mostly rendering) time, and job-splitting overhead. In comparison, prior Totems [1] training takes approximately 5 hours on one NVIDIA GeForce RTX 2080 Ti (which is most comparable to our resolution 256/512 runtimes, assuming equal parallelization across 16 GeForce GPUs).

- Additionally, as high as 20x speedup when we sparsify the training data (negligible performance difference) to a granularity as low as *g=0.05* (linear with runtime)

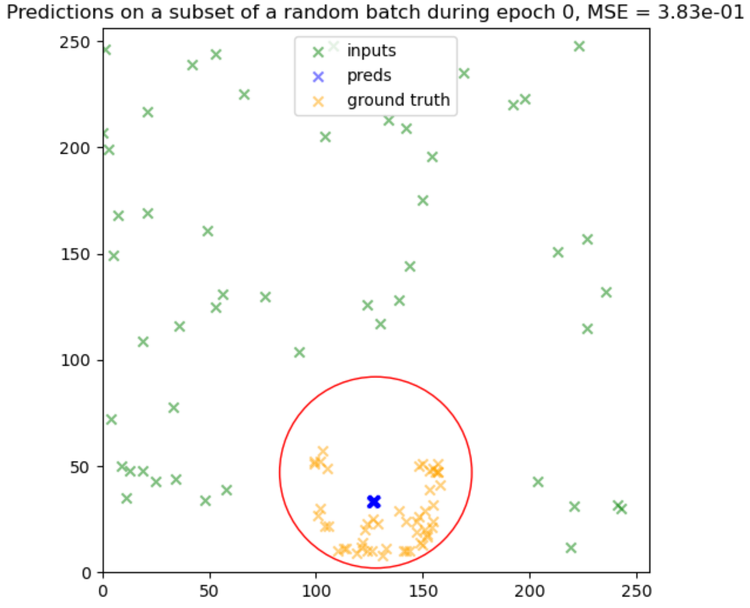
Results

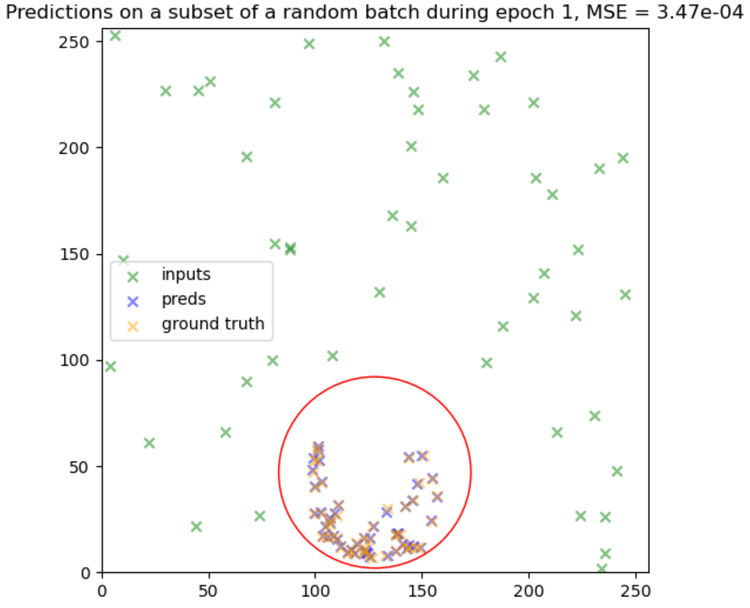
train-test-val: 80-10-10

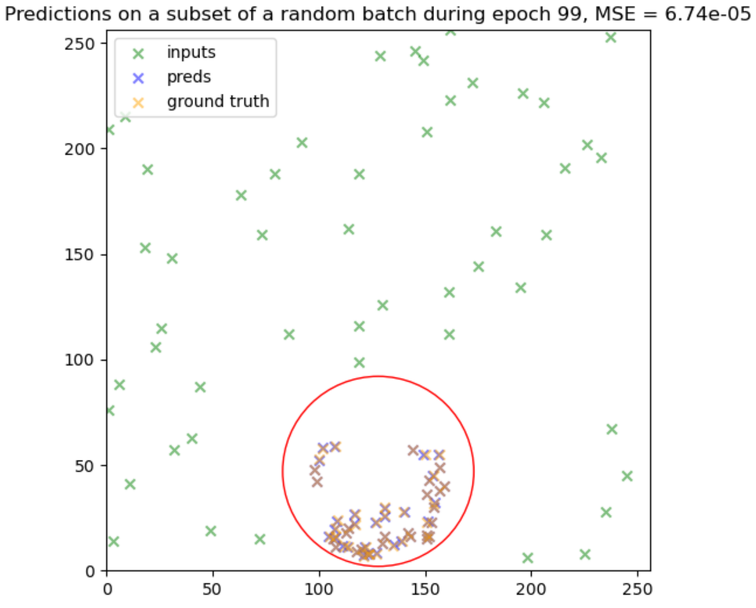
Adam optimizer



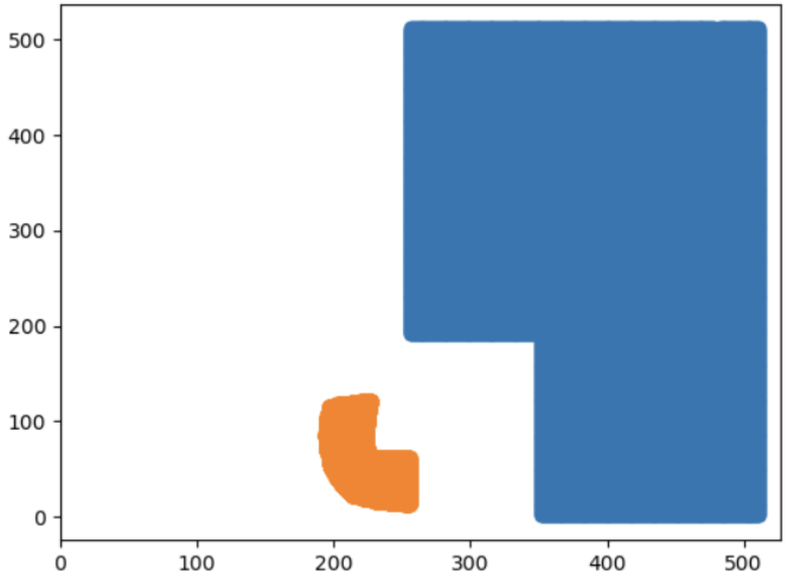
8 layers and skip connection



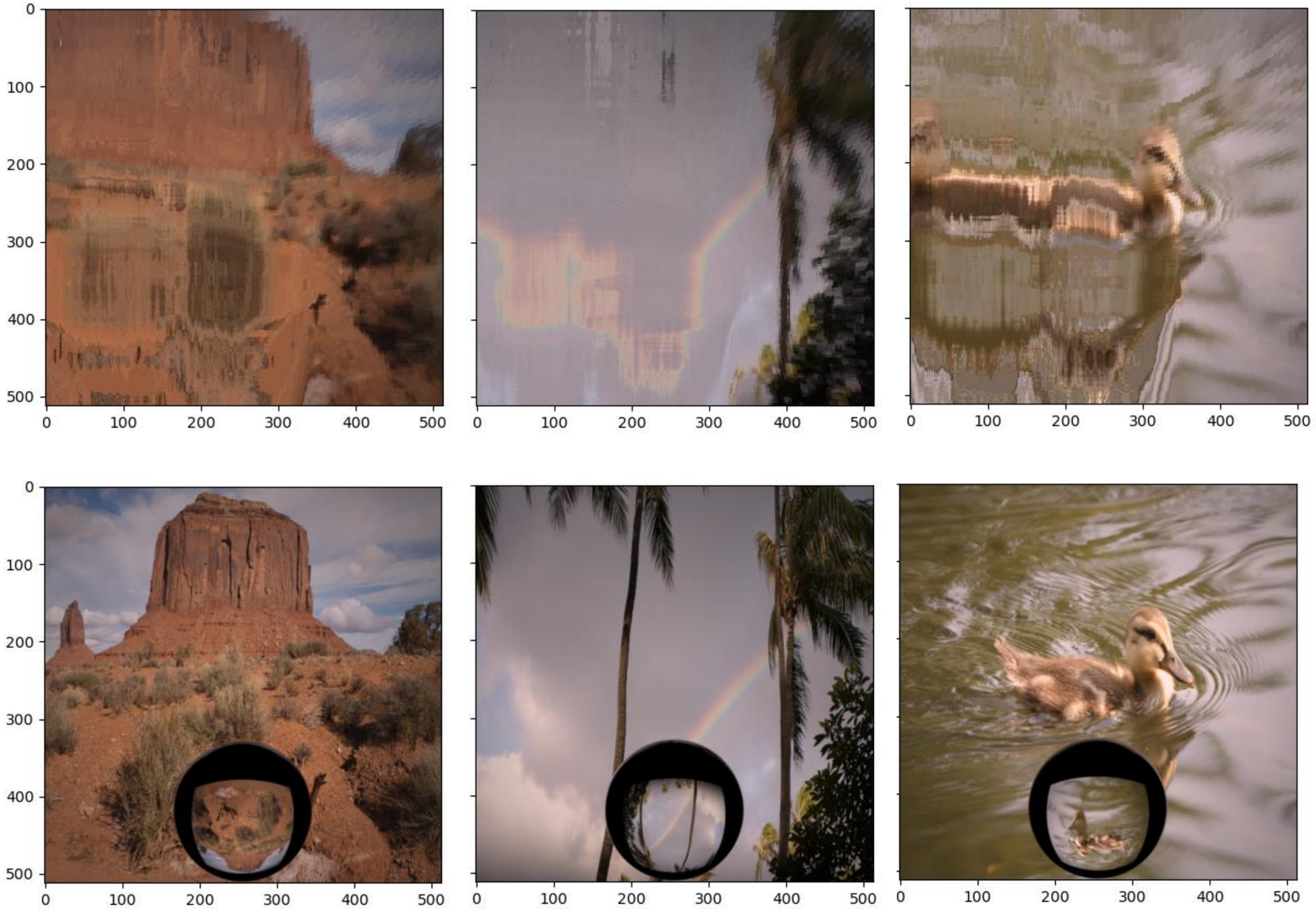




Verifying no data leakage

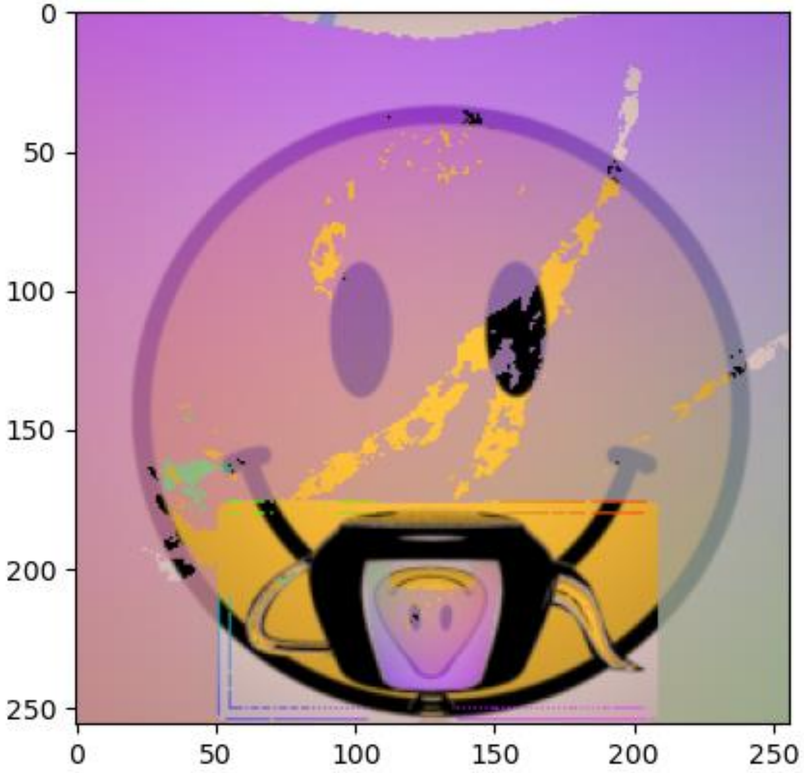


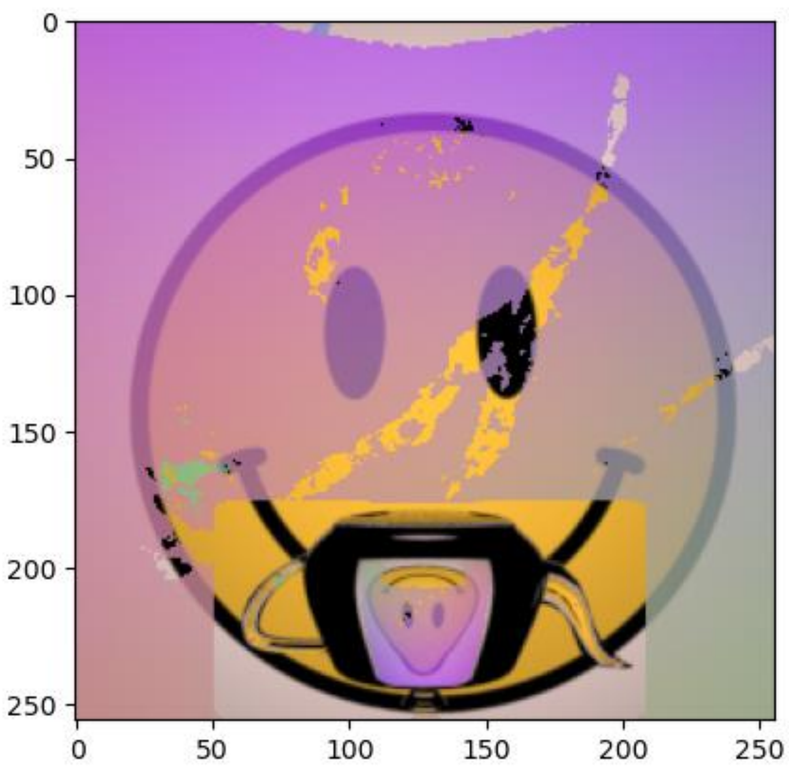
Train only right half of dataset

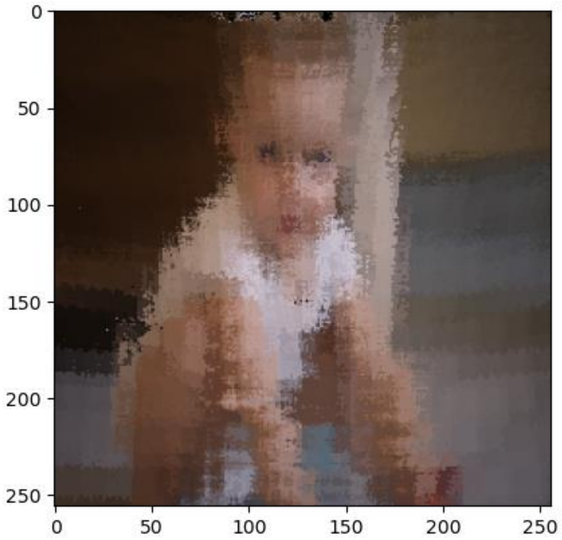


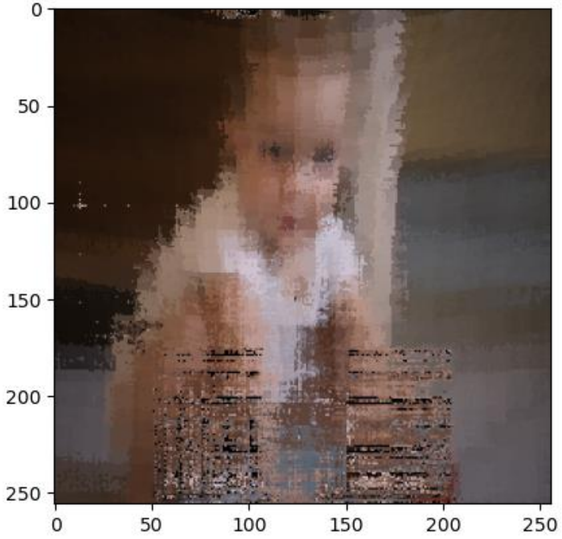
Sensitivity analysis

Adding noise to the region of no data



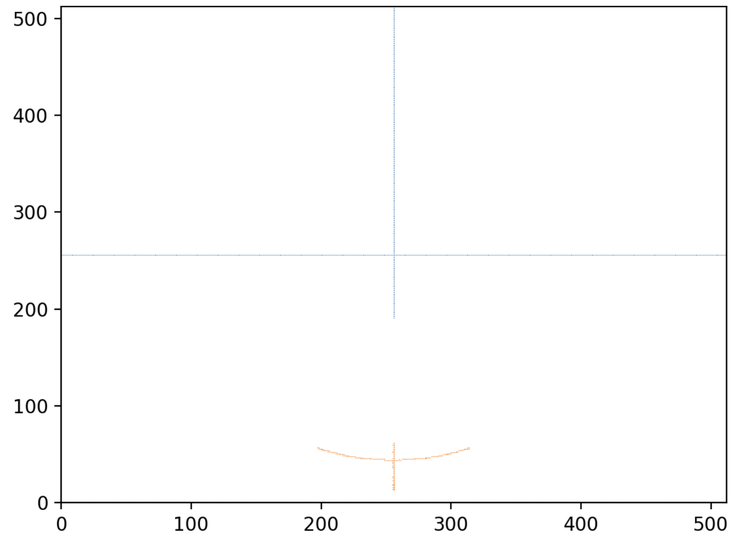


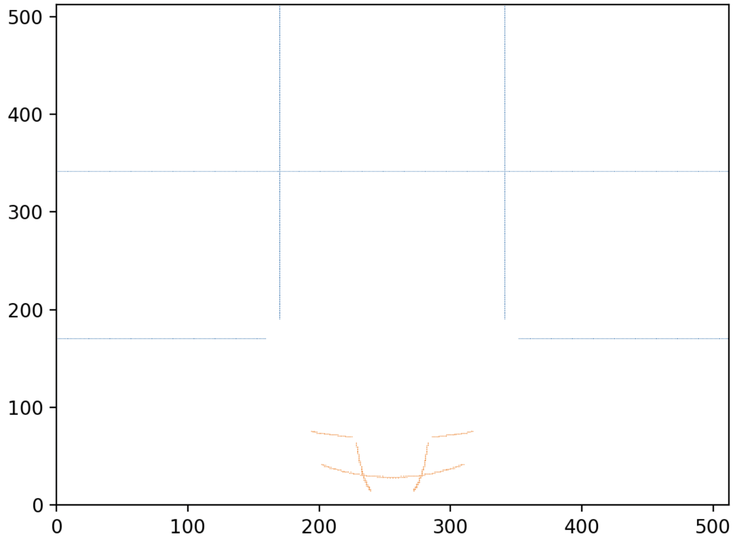


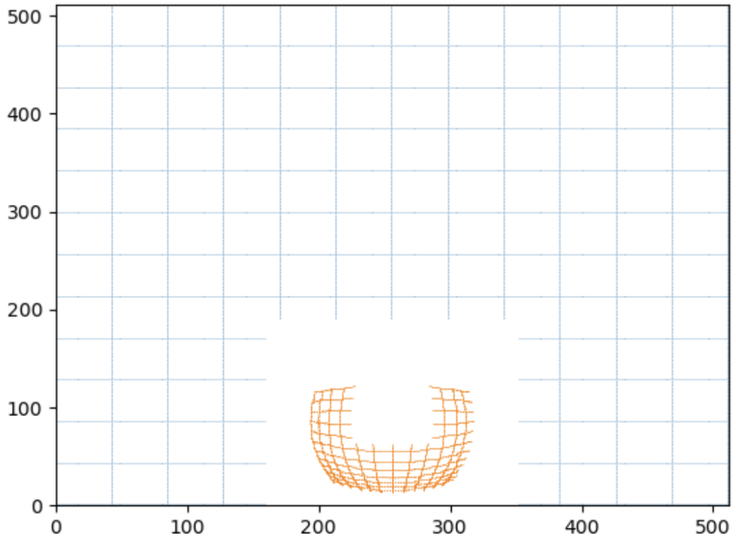


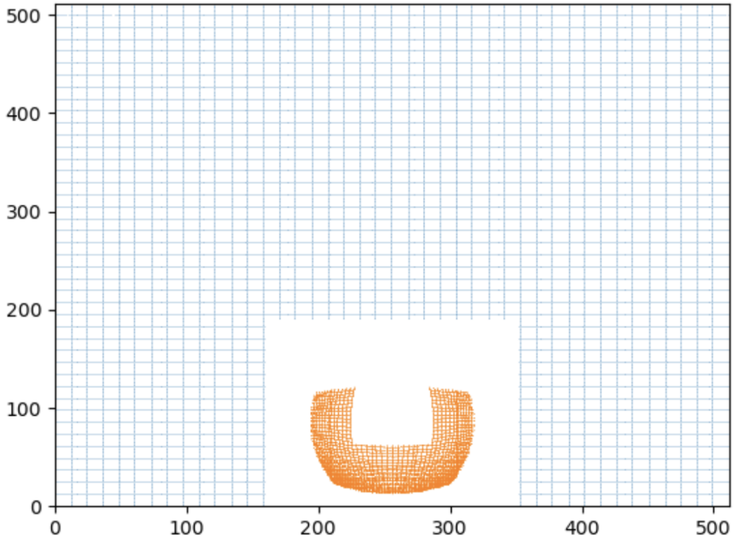
Can we train with less data?

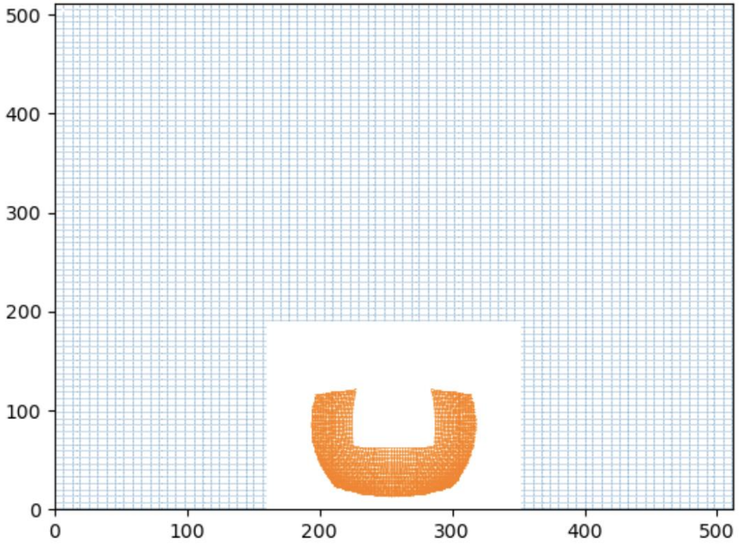
Subsample at a lower effective sweep granularity



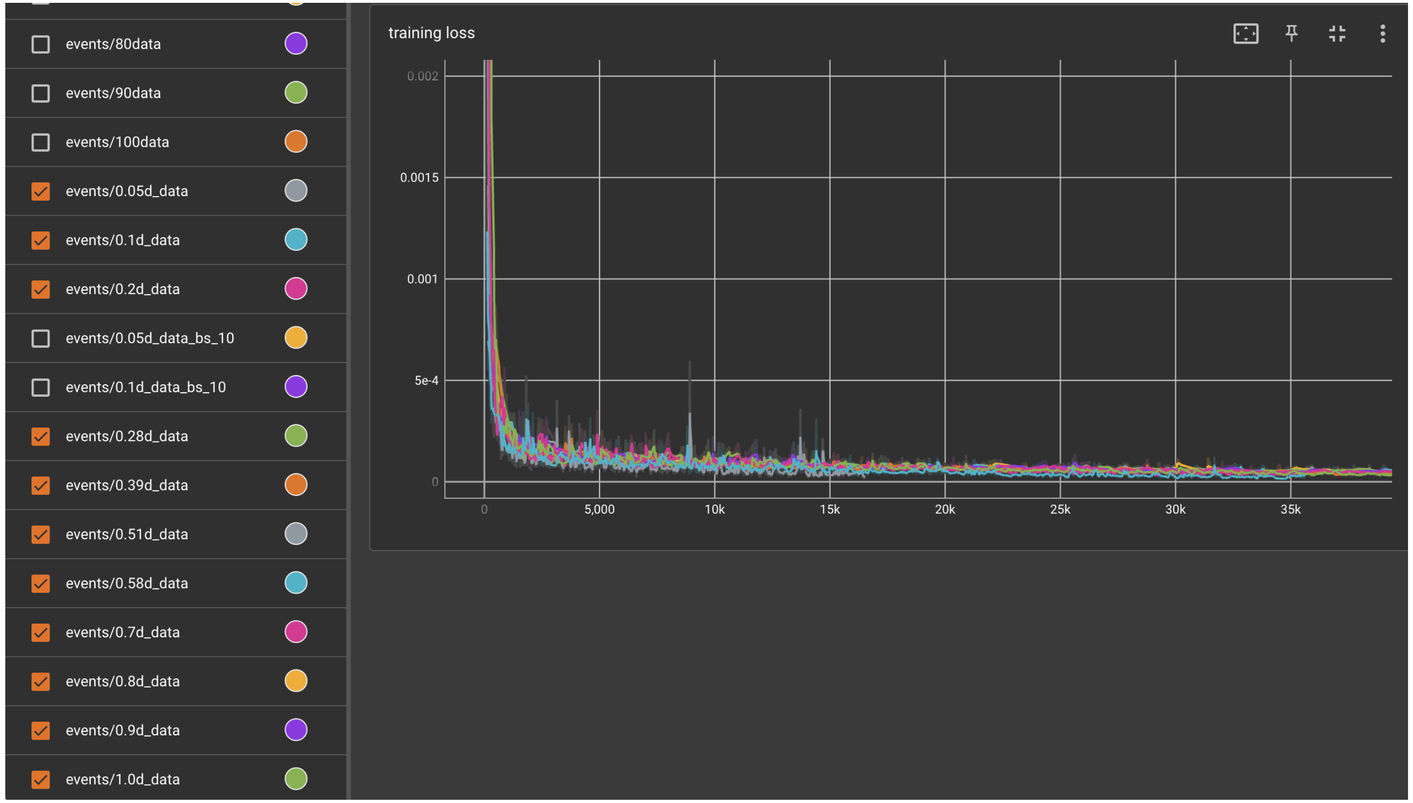




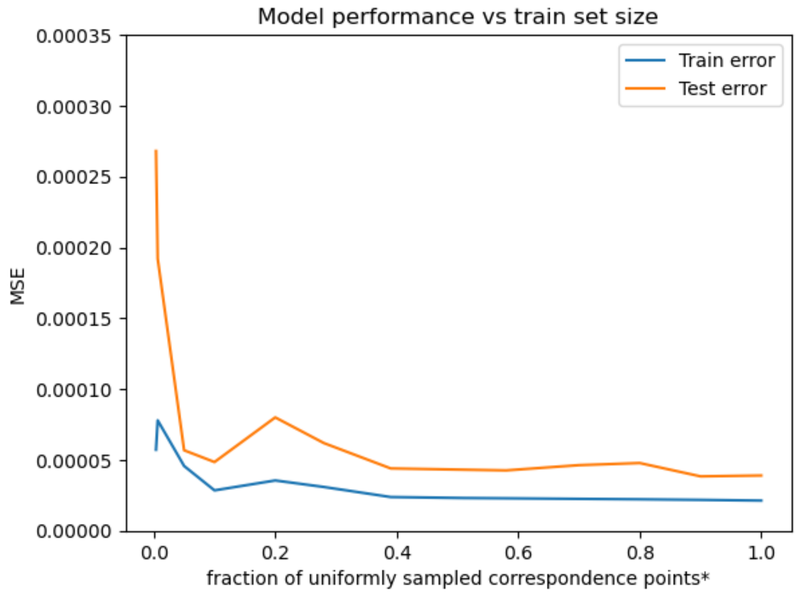




MLP Training Results

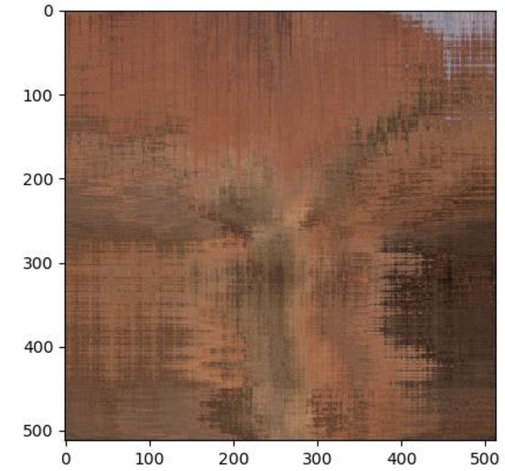


Training curves for the following fractions of full dataset: (rounded to 1 sig fig) [0.004, 0.007, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 1.0]

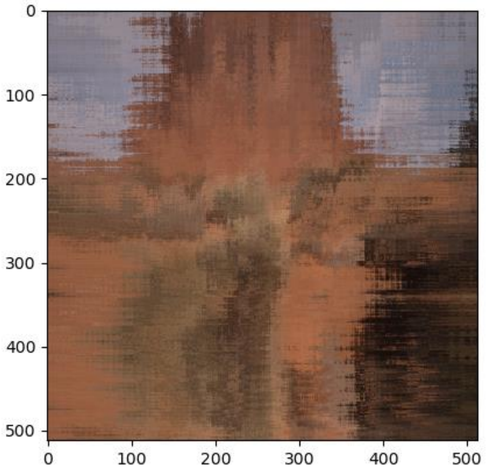


Degradation of error at granularities < 0.05

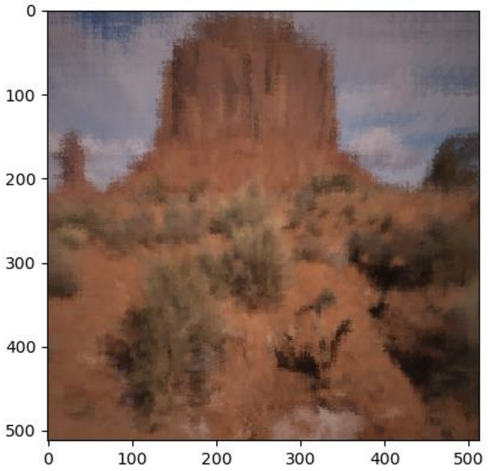
Unwarping results: qualitative



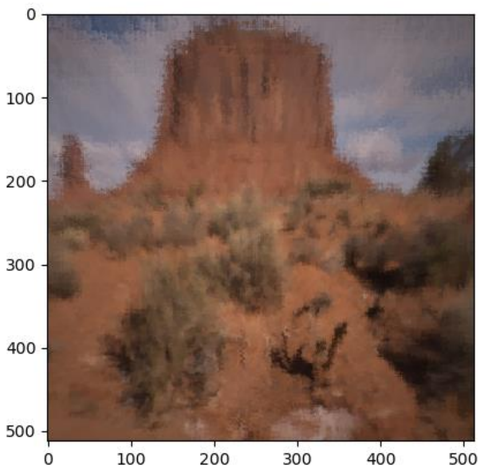
0.004



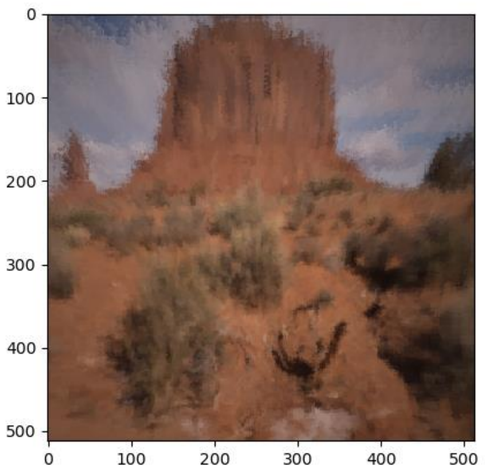
0.007



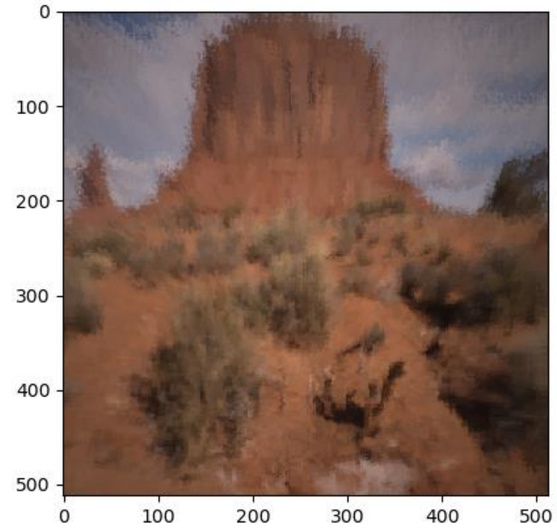
0.05



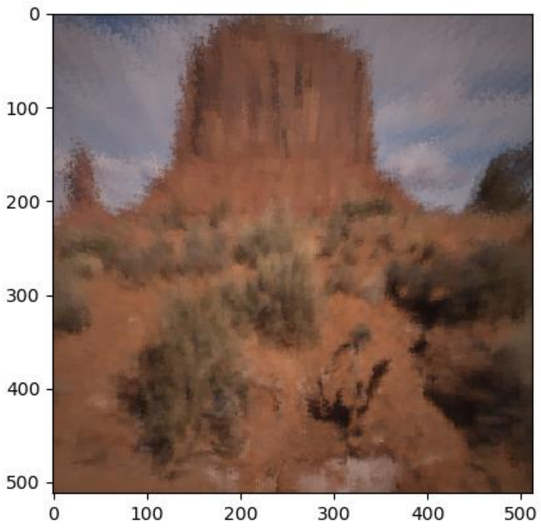
0.1



0.2

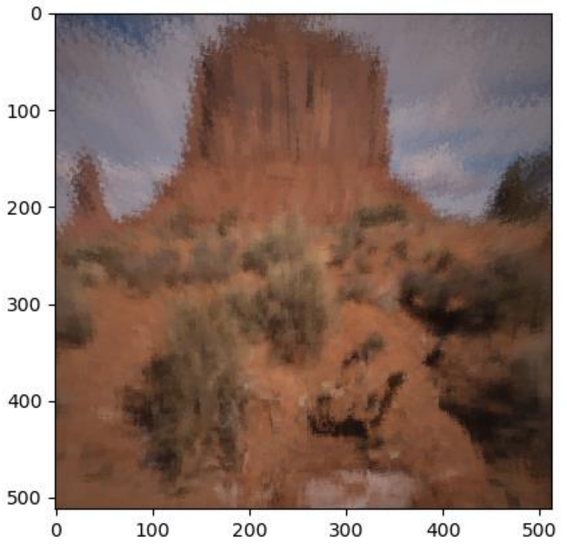


0.3

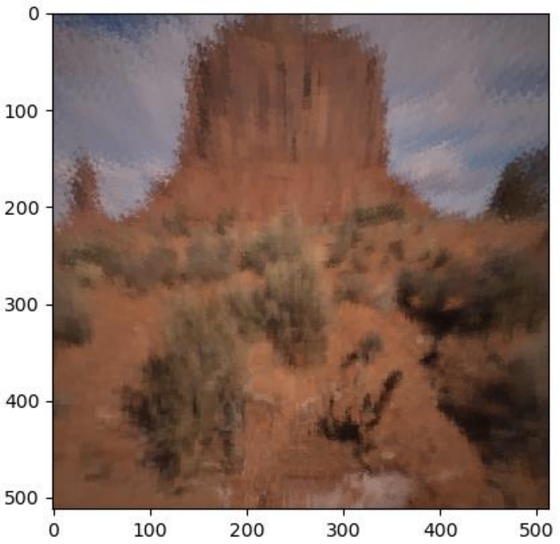


0.4

...

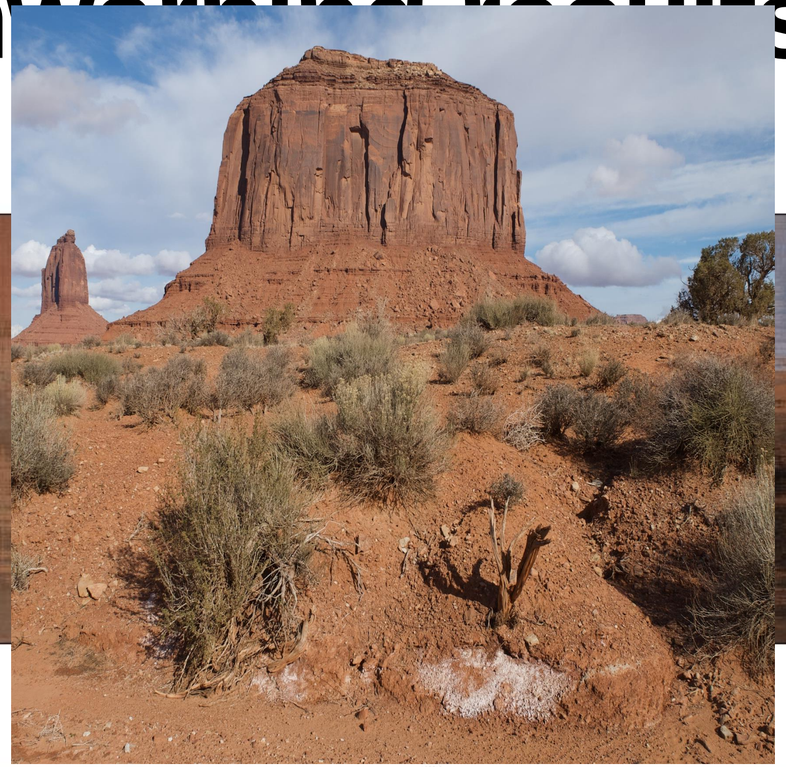


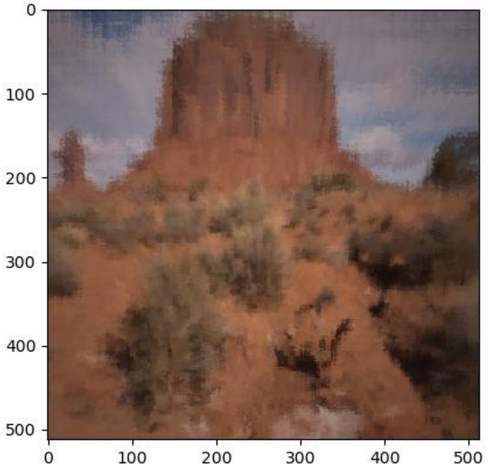
0.9



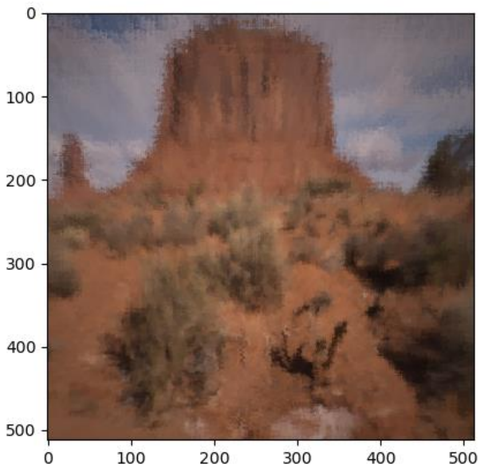
1.0

Unwarping results: qualitative

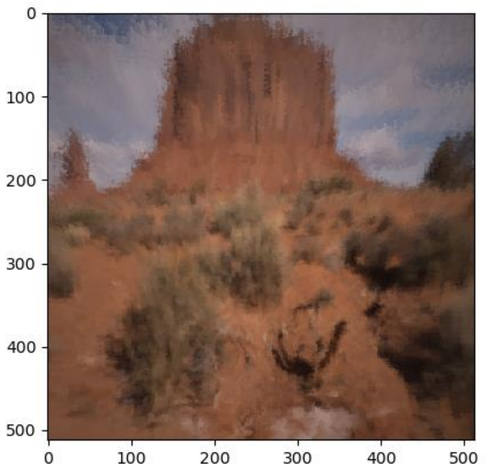




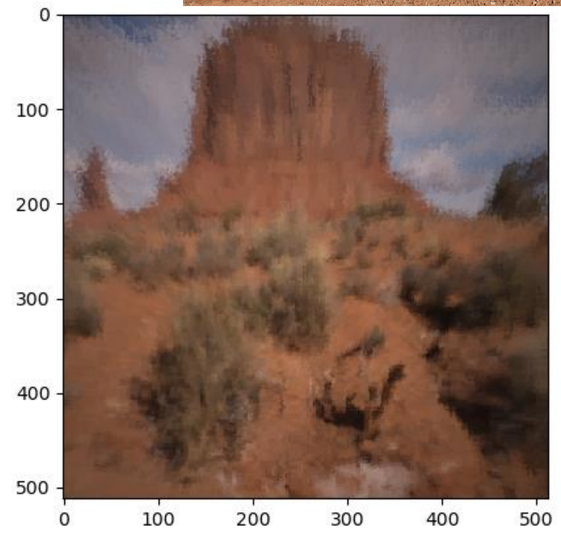
0.05



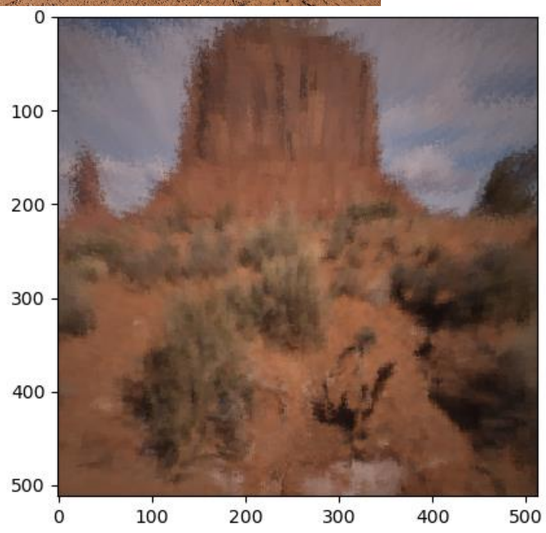
0.1



0.2

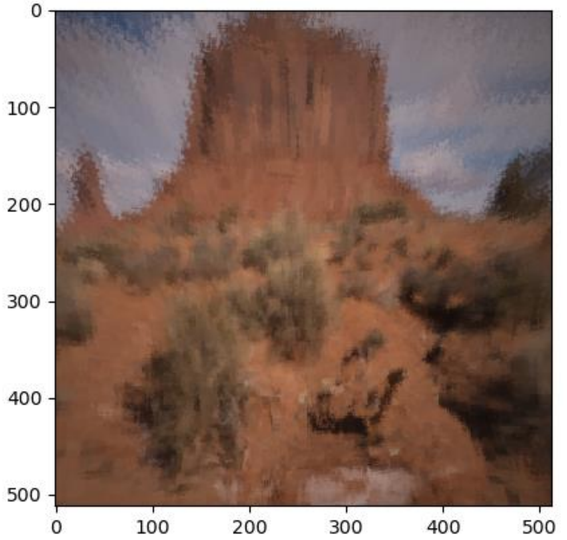


0.3

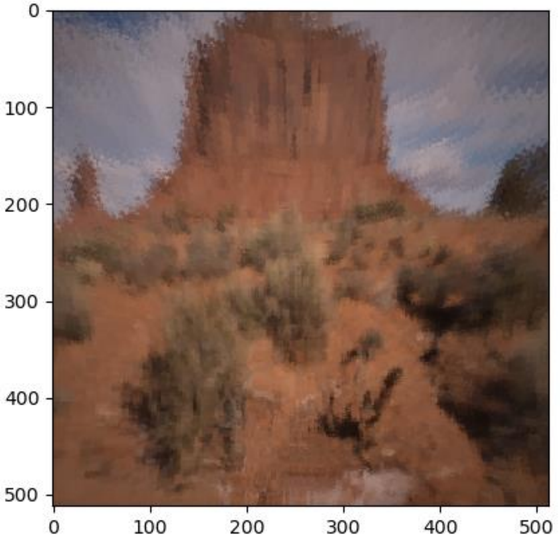


0.4

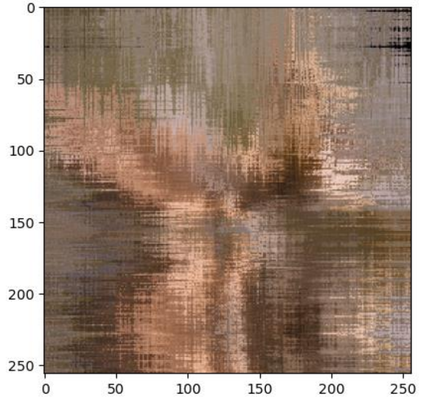
...



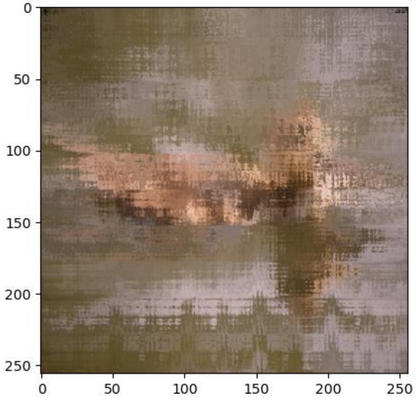
0.9



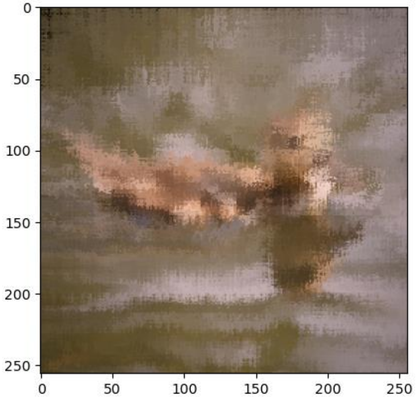
1.0



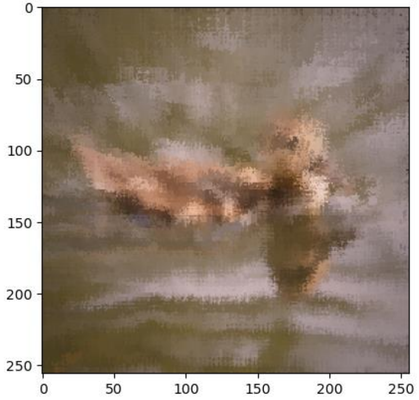
0.007



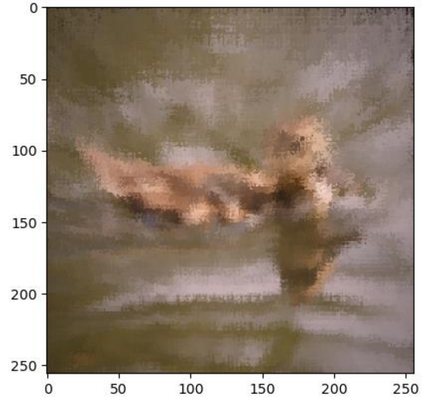
0.03



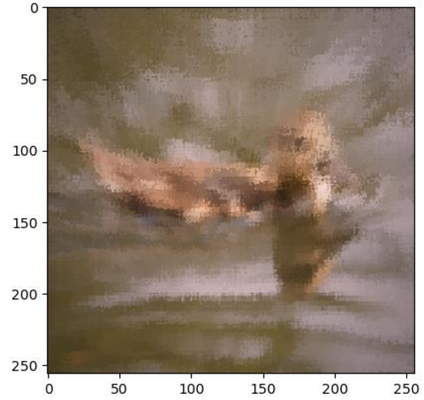
0.05



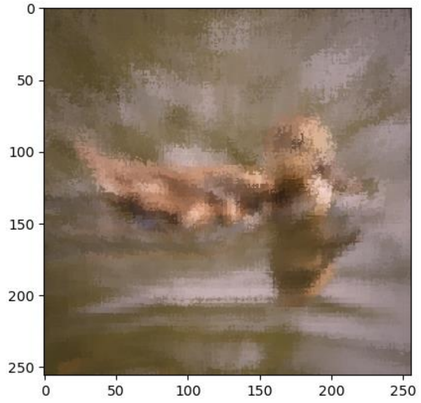
0.10



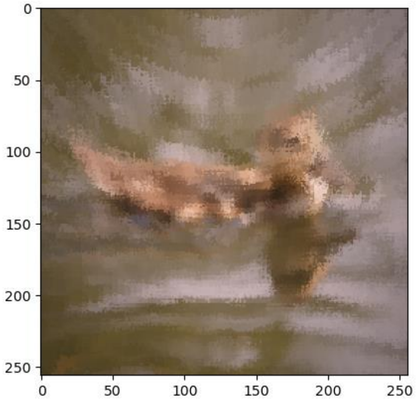
0.20



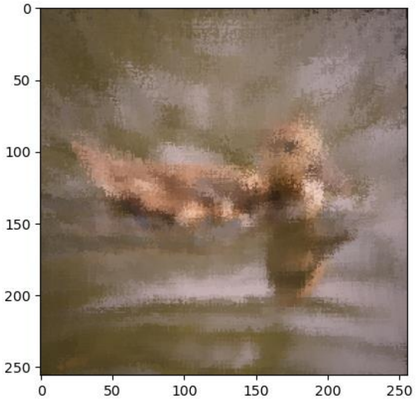
0.30



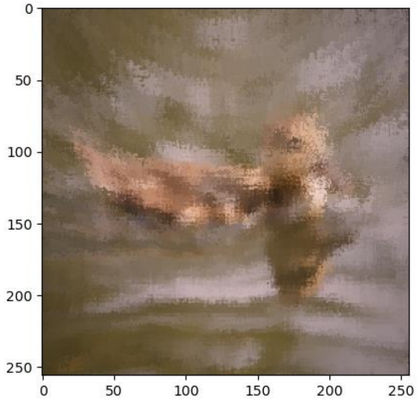
0.40



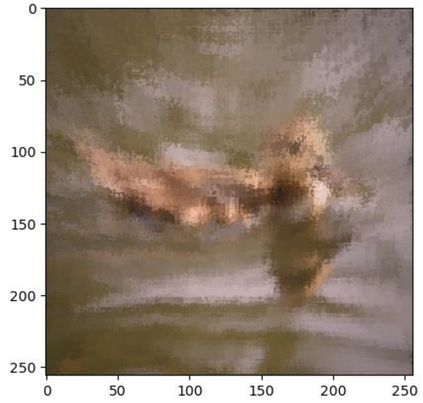
0.50



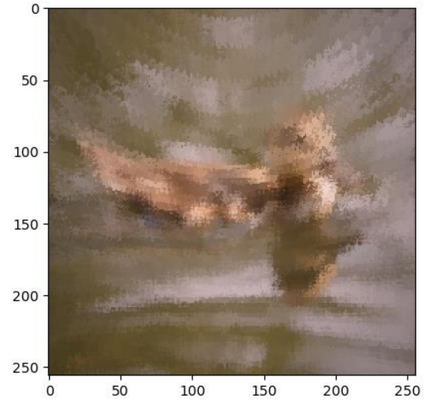
0.60



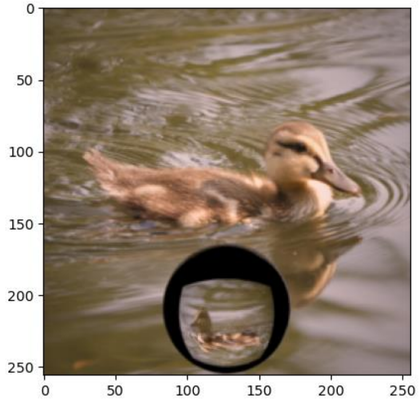
0.70



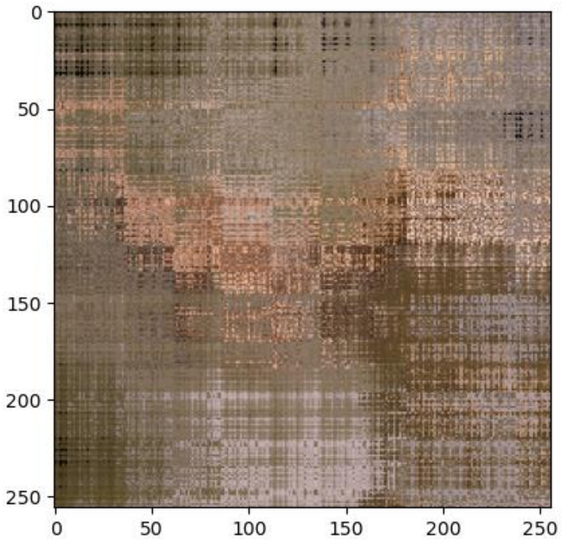
0.80



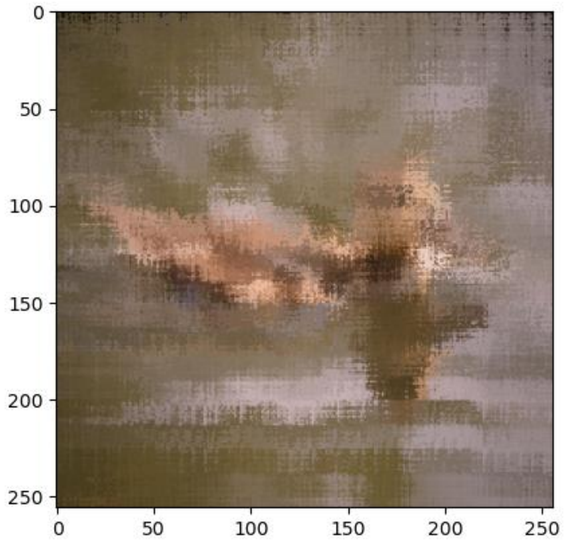
0.90



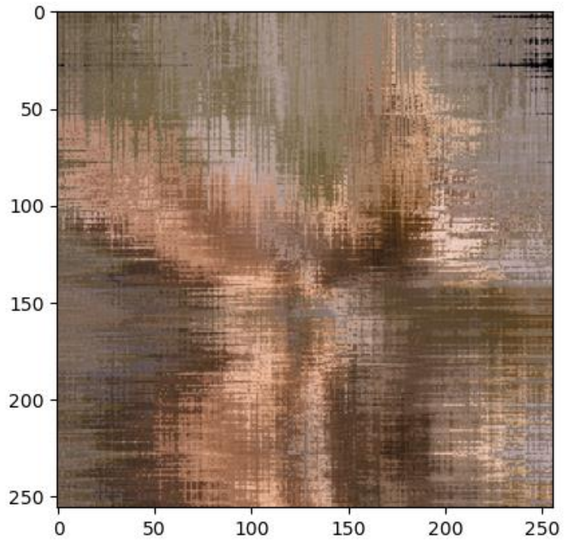
Original



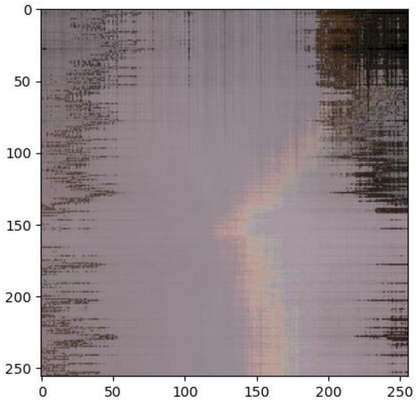
0.001, random (uniform P)



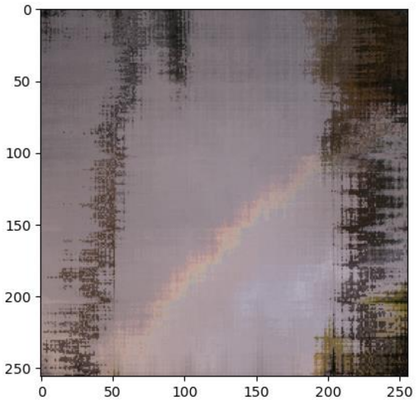
0.007, random (uniform P)



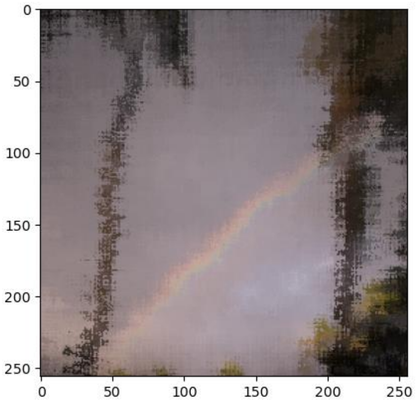
0.007, uniform lines



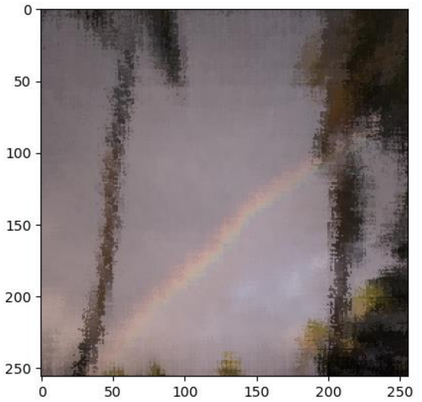
0.007



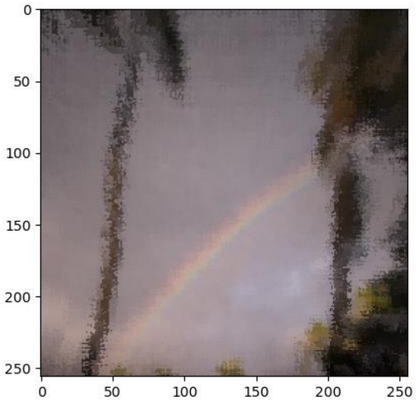
0.03



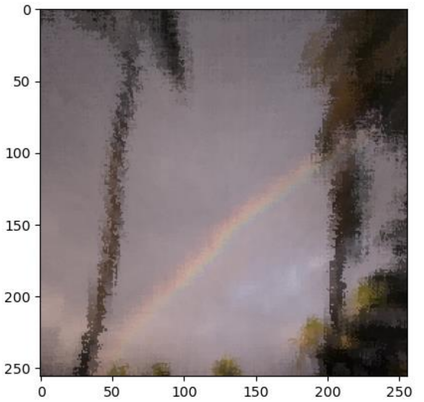
0.05



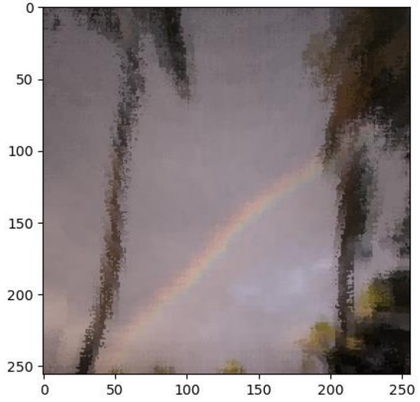
0.10



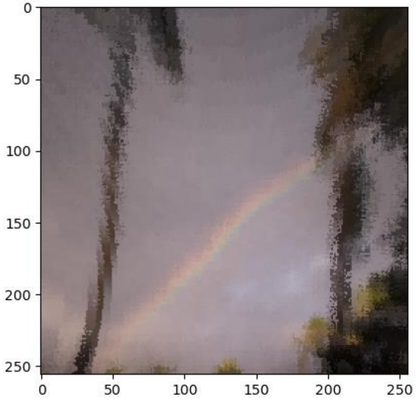
0.20



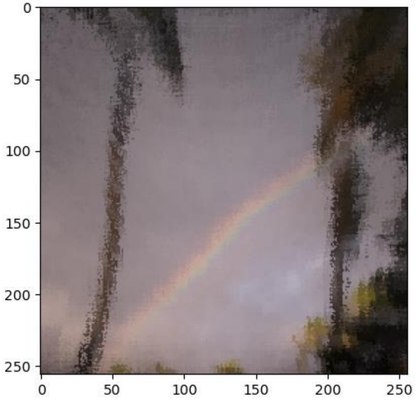
0.30



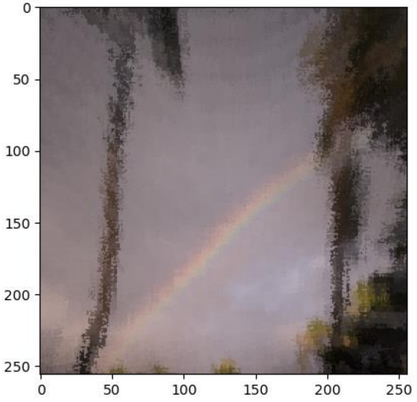
0.40



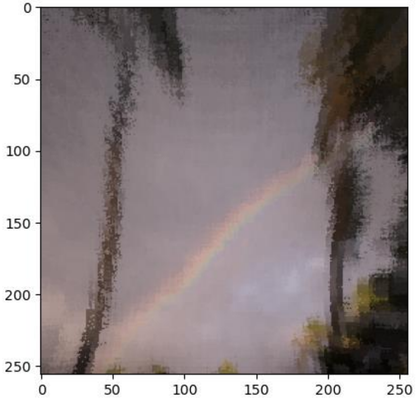
0.50



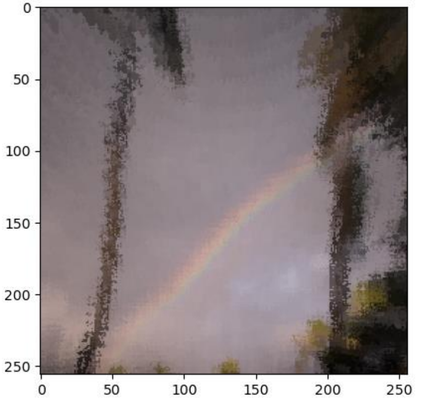
0.60



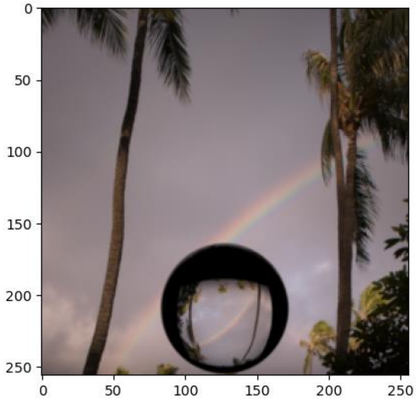
0.70



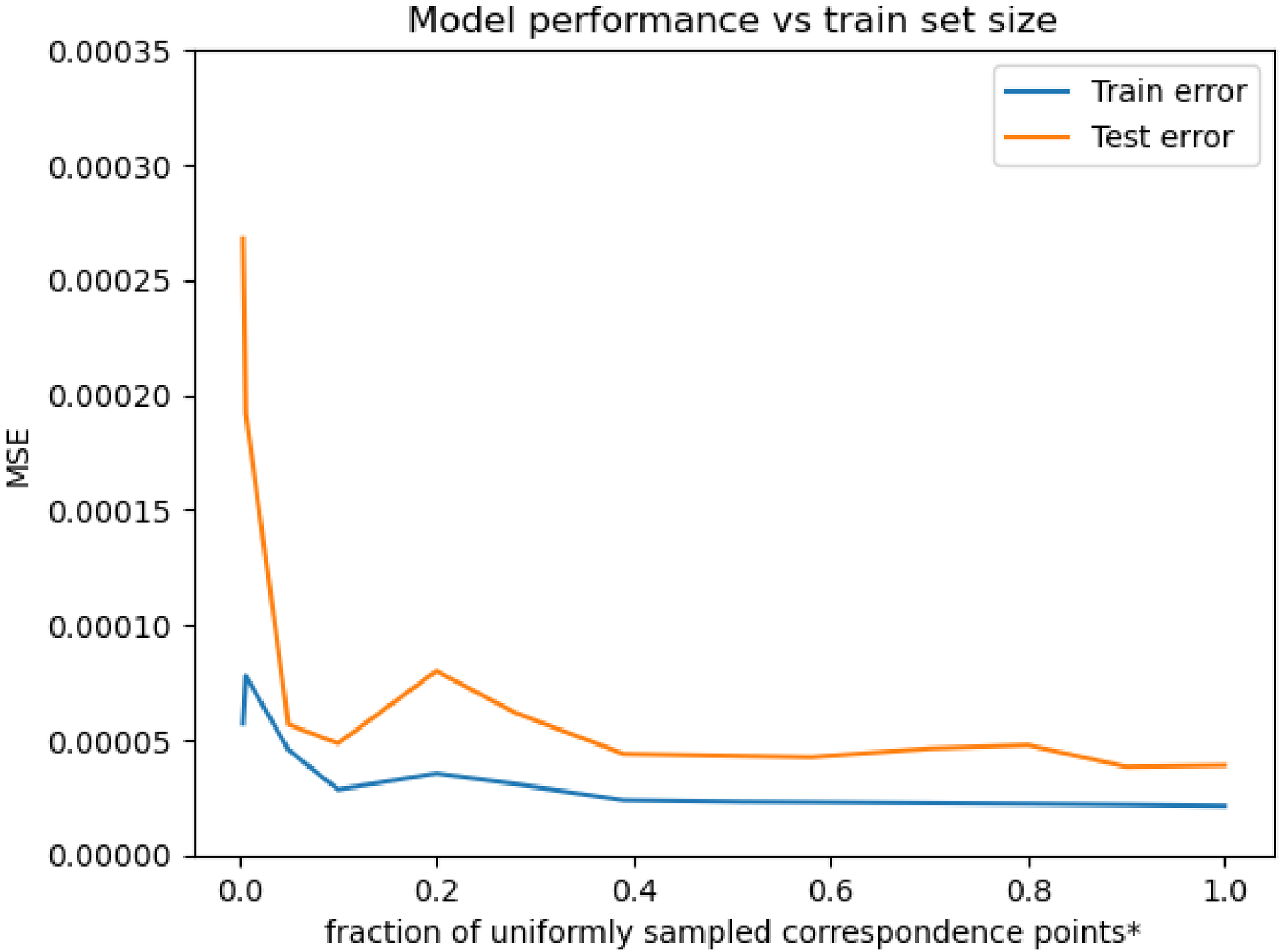
0.80



0.90

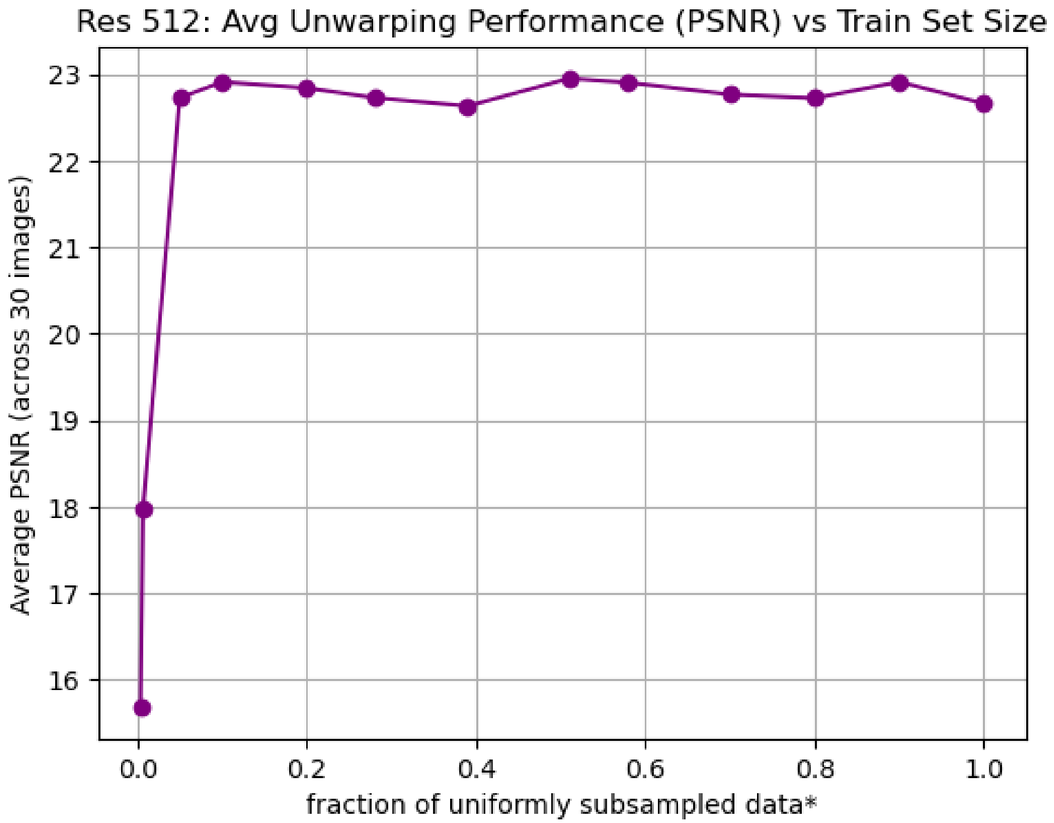


Original


Model performance vs train set size
Train error
Test error
MSE
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0.0
0.2
0.4
0.6
0.8
1.0
fraction of uniformly sampled correspondence points*

Unwarping results: quantitative

Psnr vs granularity



Method

For a desired reconstruction resolution,

1. Render the scene with the totem and camera placed according to the same settings as camera calibration

2. For each pixel (i,j) in the image to be reconstructed:

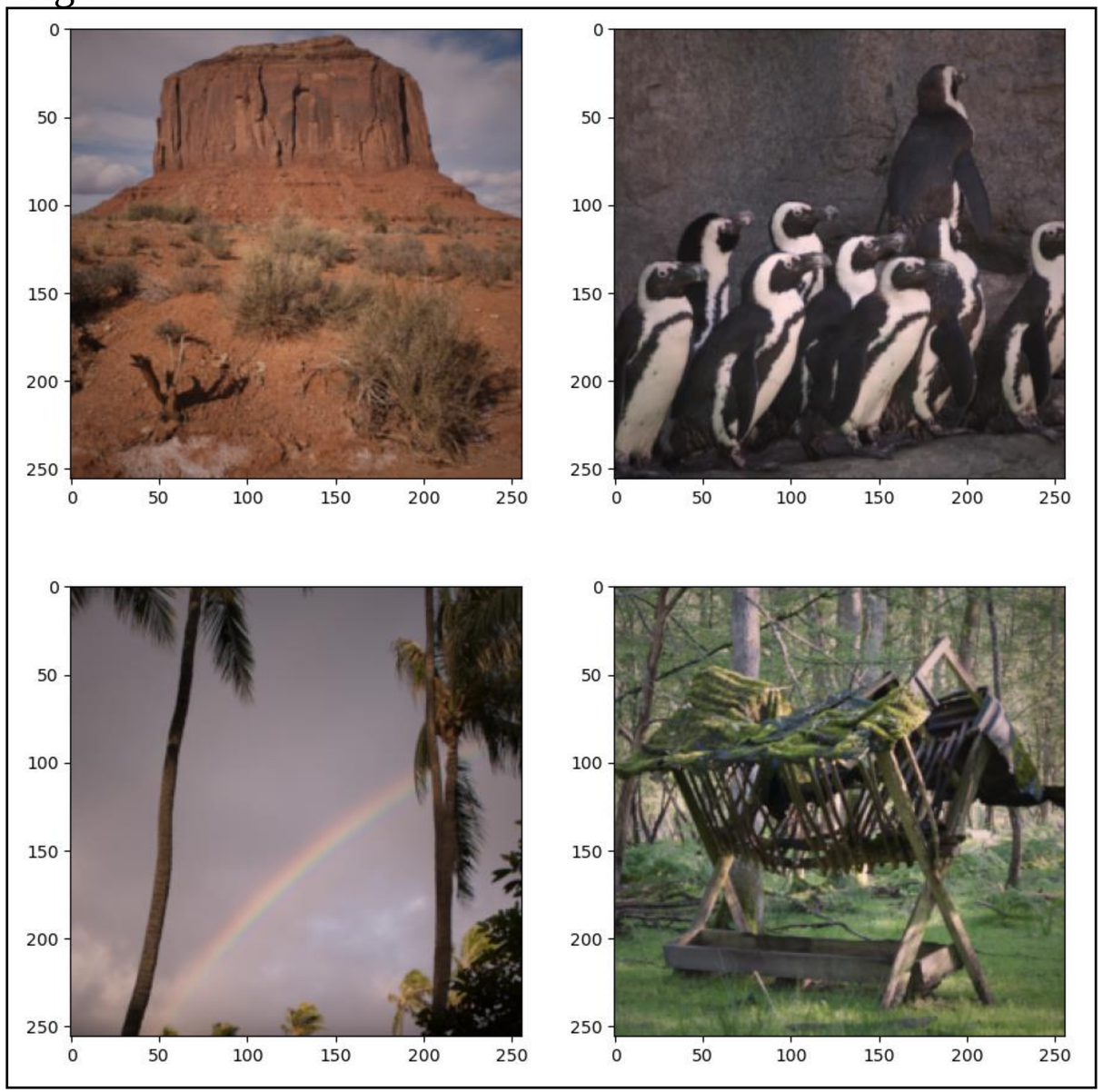
    1. Lookup the totem correspondence $(u_i, v_j)_t$ for pixel $(u_i, v_j)_c$

    2. If does not exist, obtain the correspondence from inference using the MLP

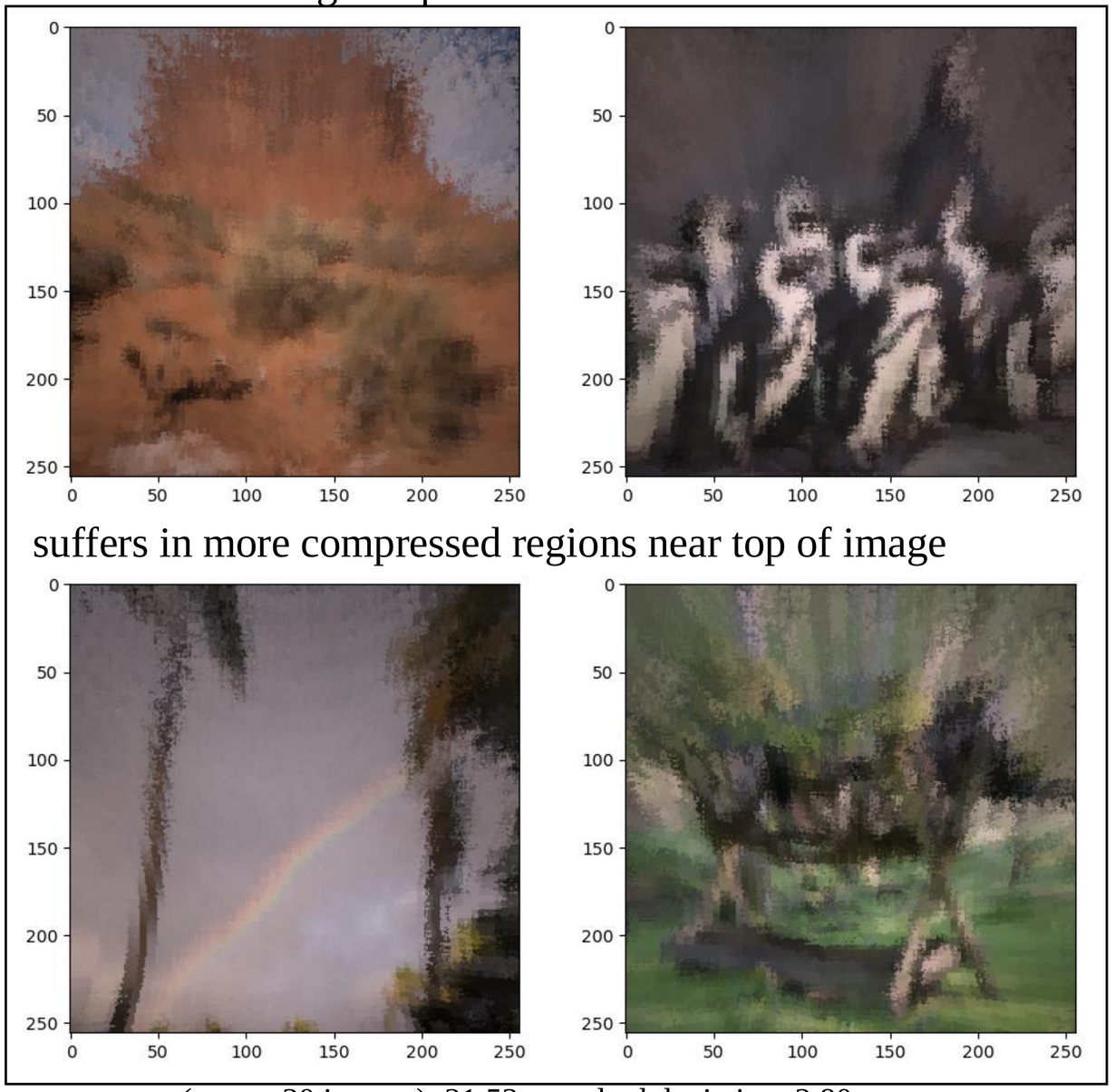
    3. Fill in the pixel (i,j)'s RGB value according to the RGB value of the totem pixel $(u_i, v_j)_t$

Unwarping results

original



reconstructed using NN preds

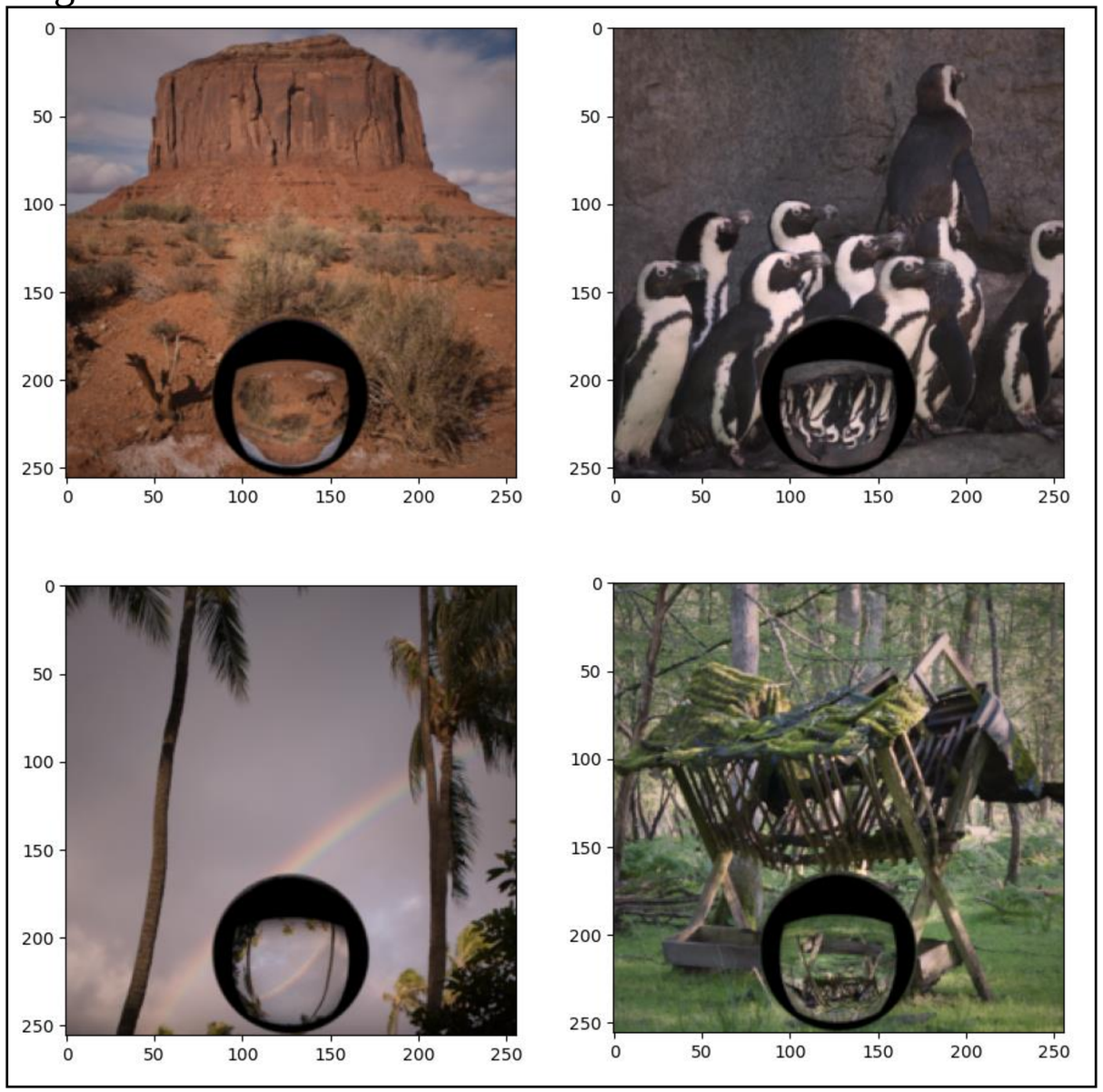


suffers in more compressed regions near top of image

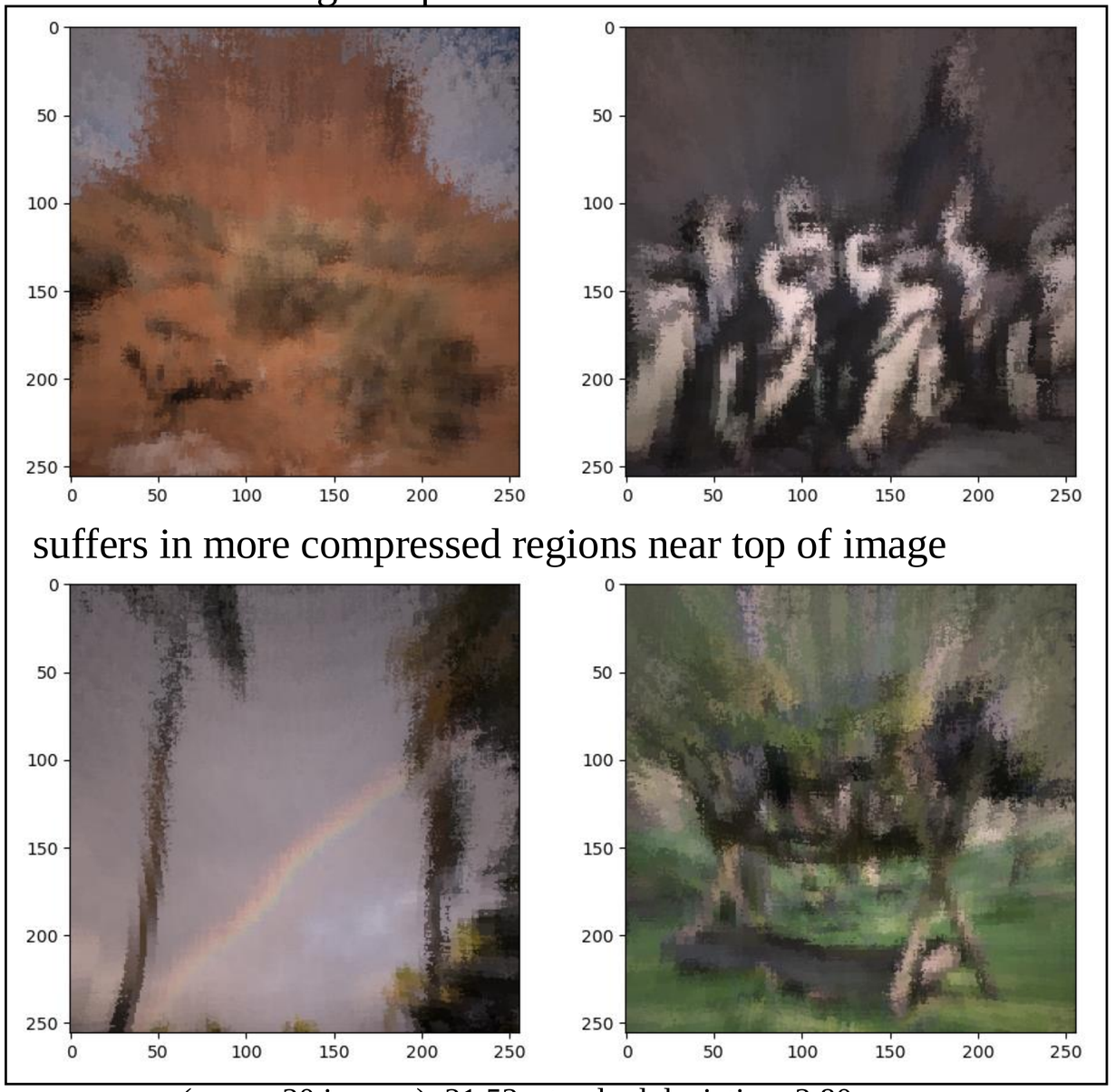
average psnr (across 30 images): 21.53; standard deviation: 2.80

Unwarping results

original



reconstructed using NN preds

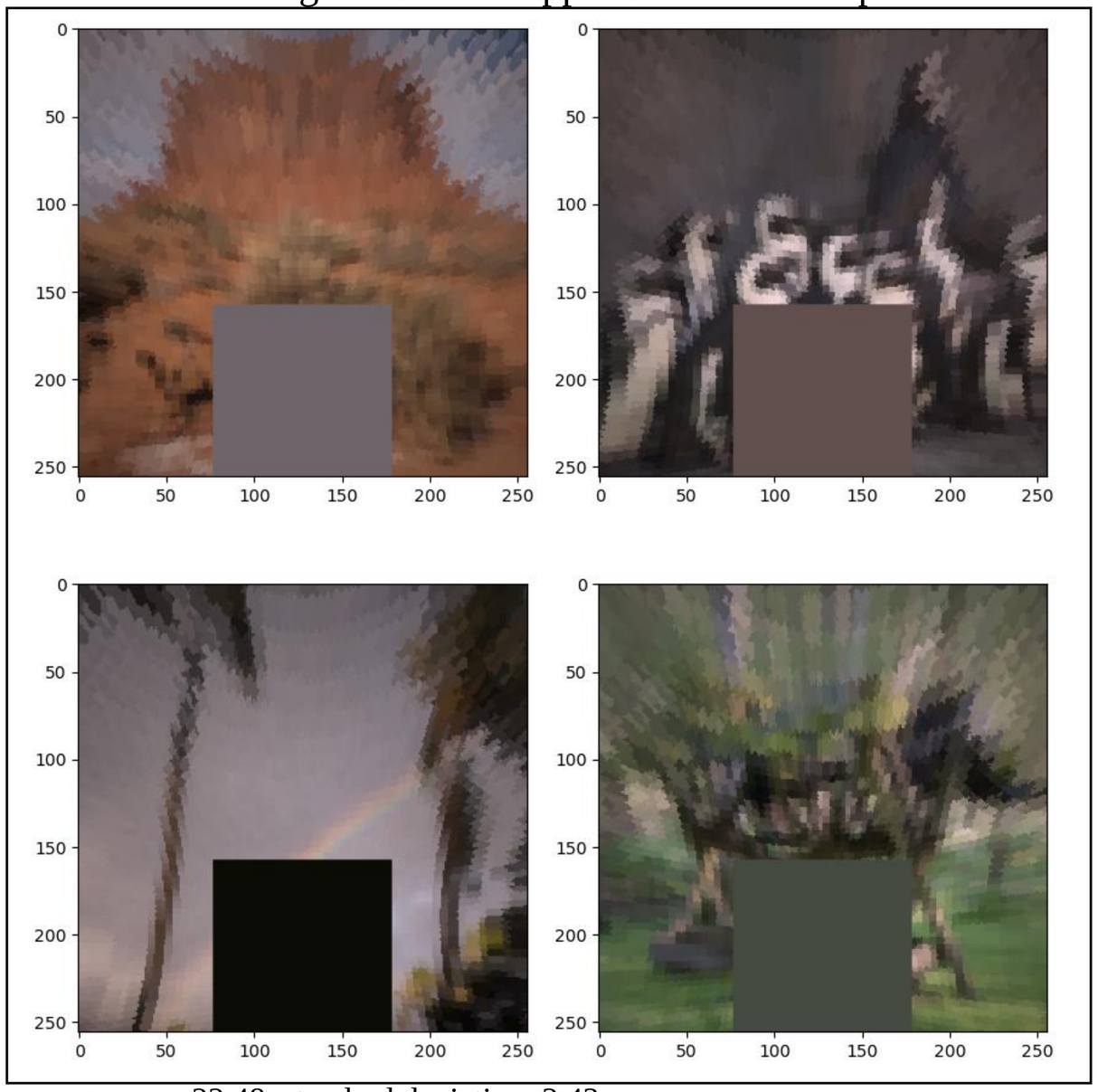


suffers in more compressed regions near top of image

average psnr (across 30 images): 21.53; standard deviation: 2.80

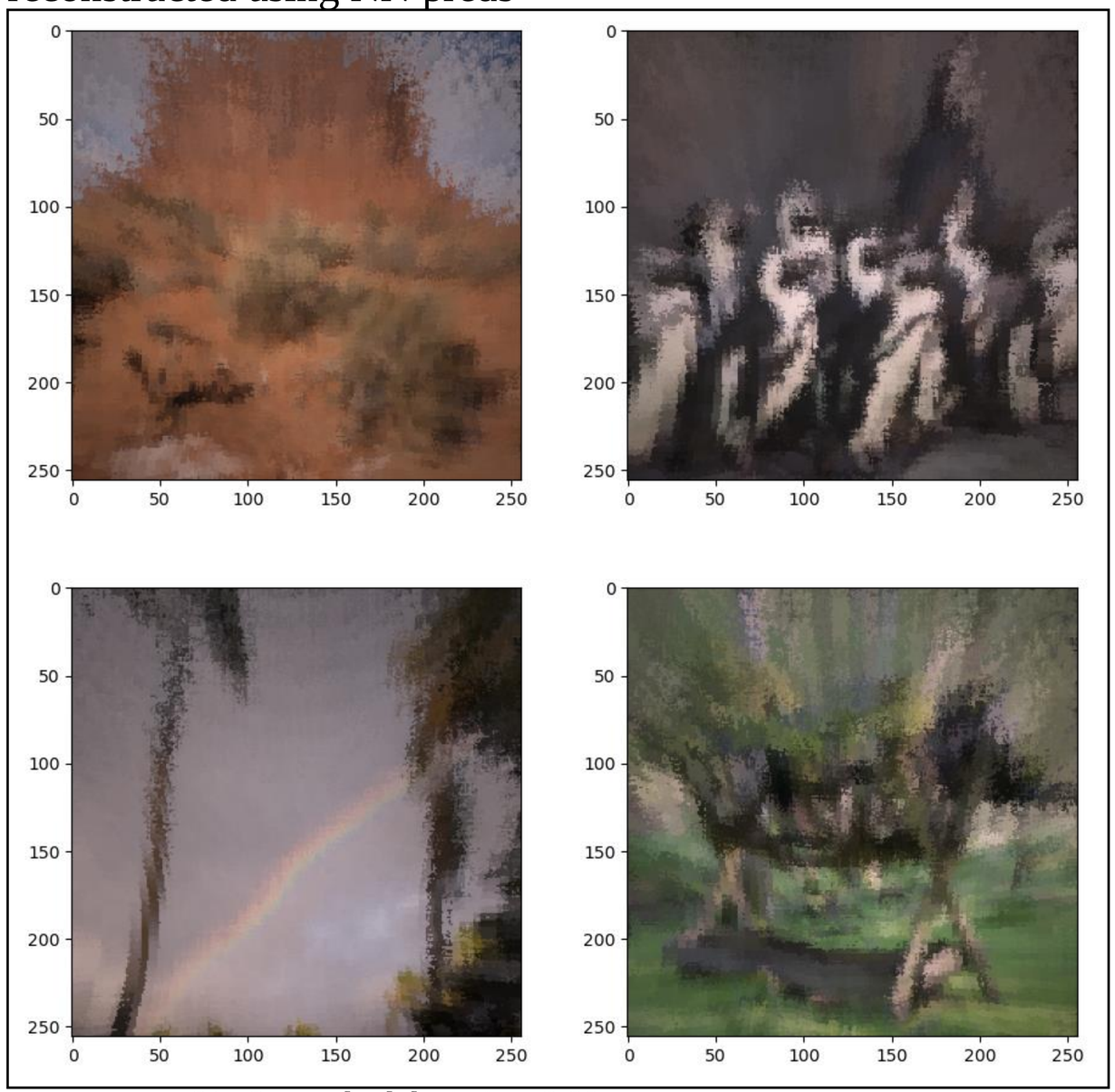
Comparing inference against dataset

reconstructed using dataset - an upper bound for NN performance



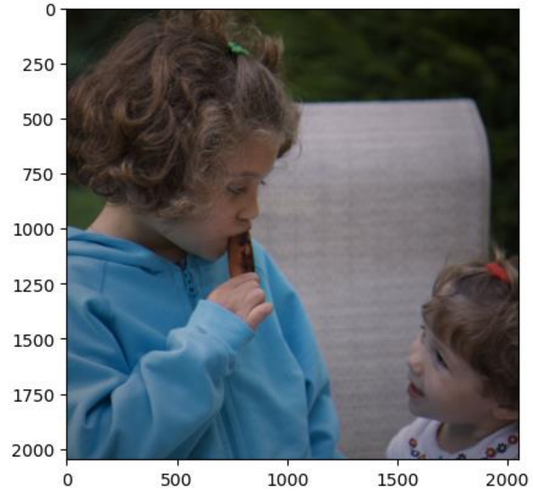
average psnr: 22.48; standard deviation: 3.43

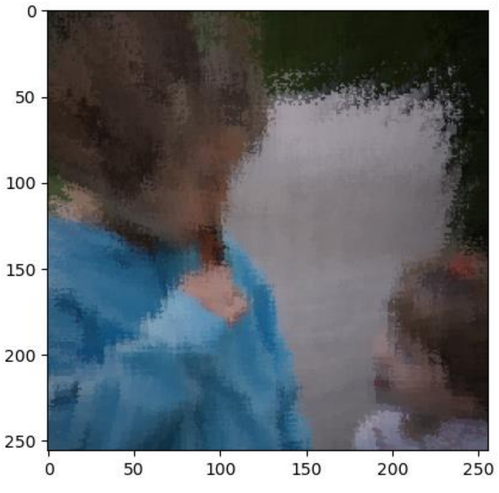
reconstructed using NN preds

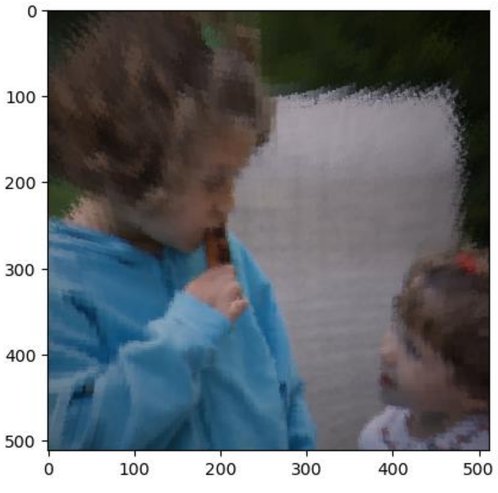


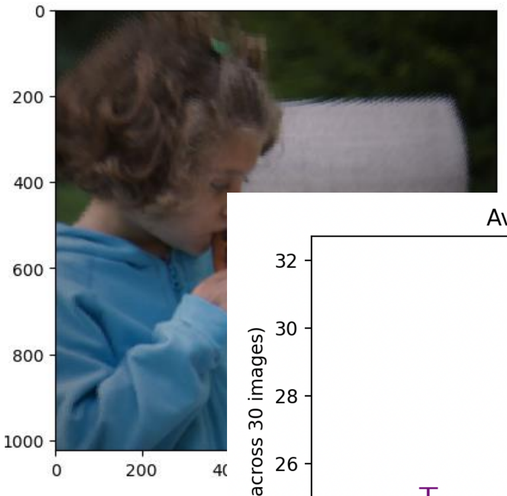
average psnr: 21.53; standard deviation: 2.80

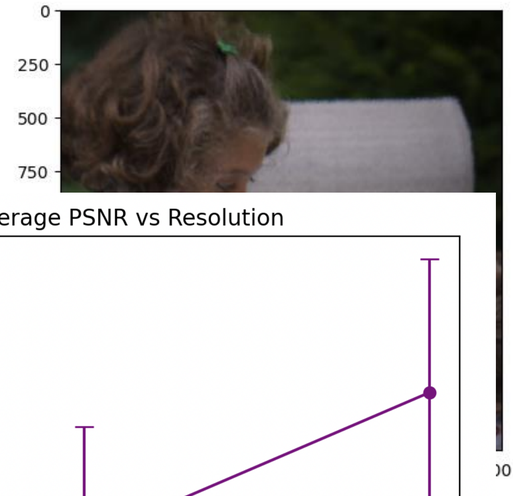
Trying higher resolutions

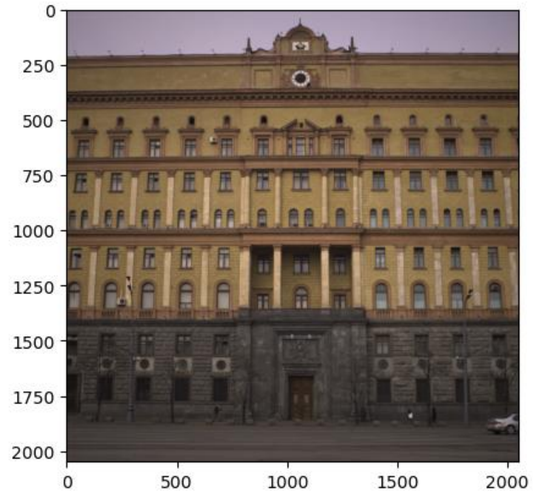


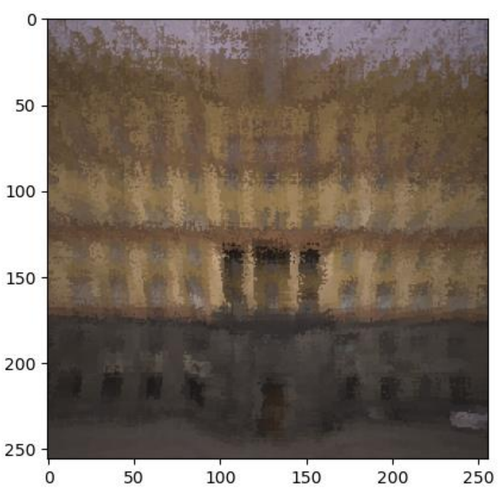


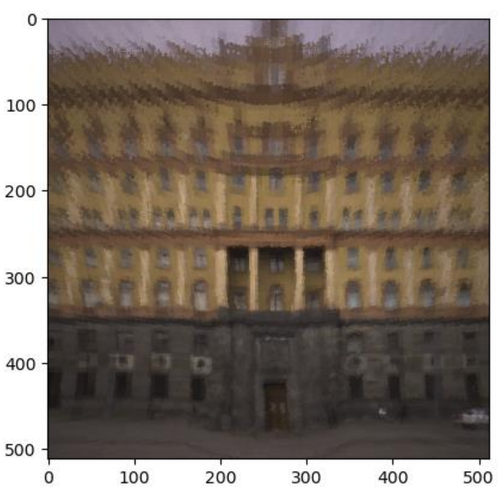


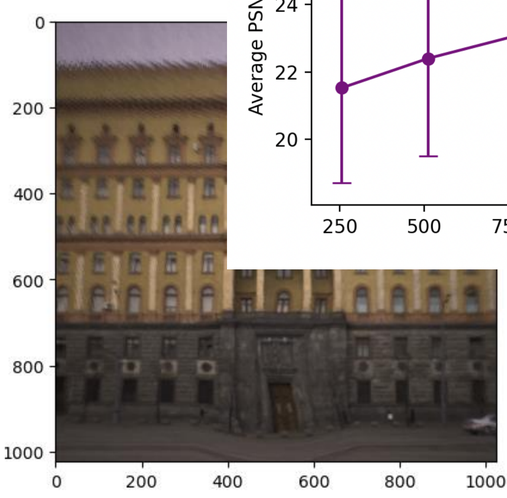


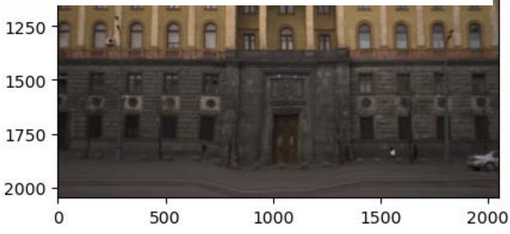






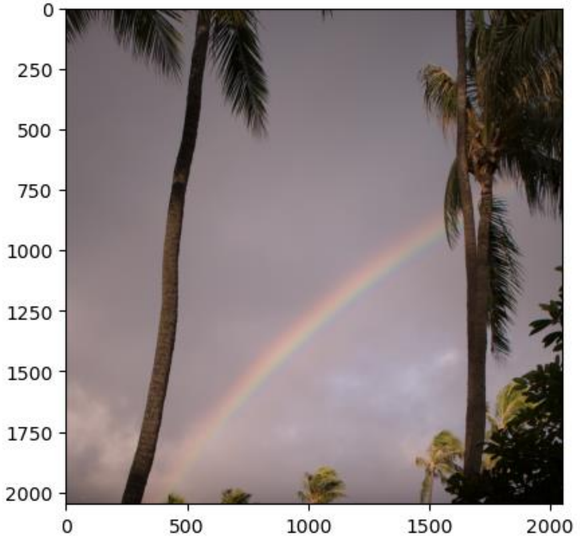


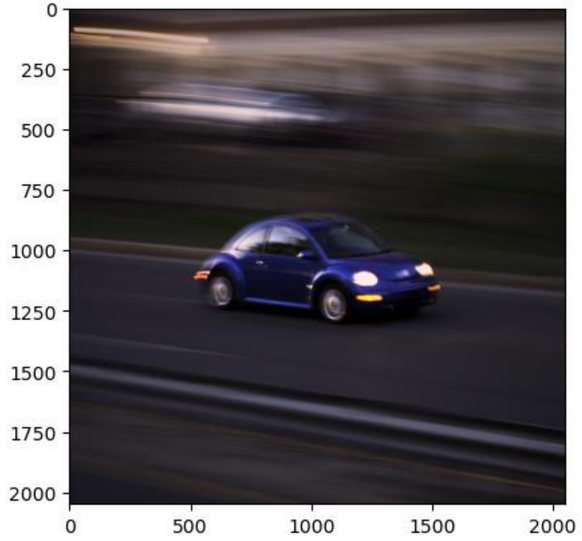


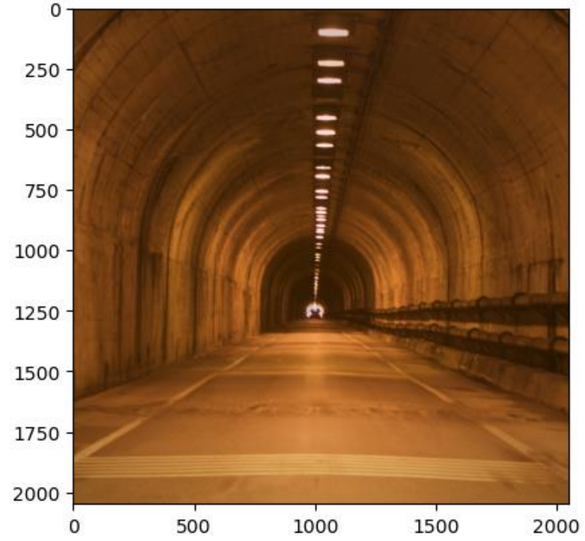


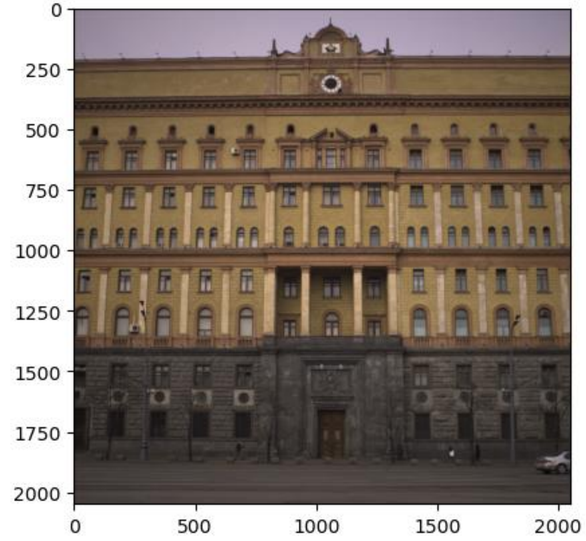
Original 256 512 1024 2048

Original Images (rendered at 2048)

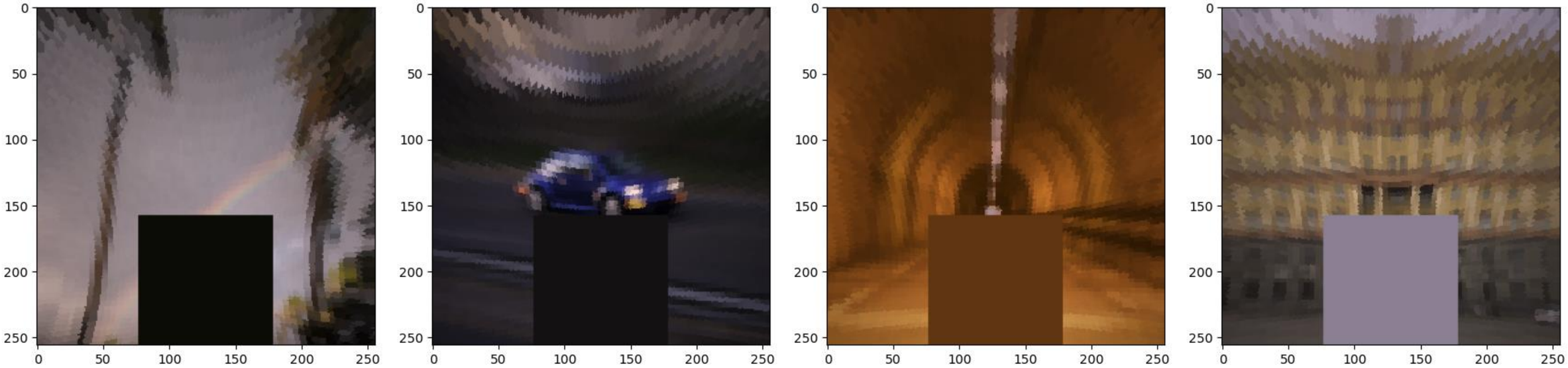




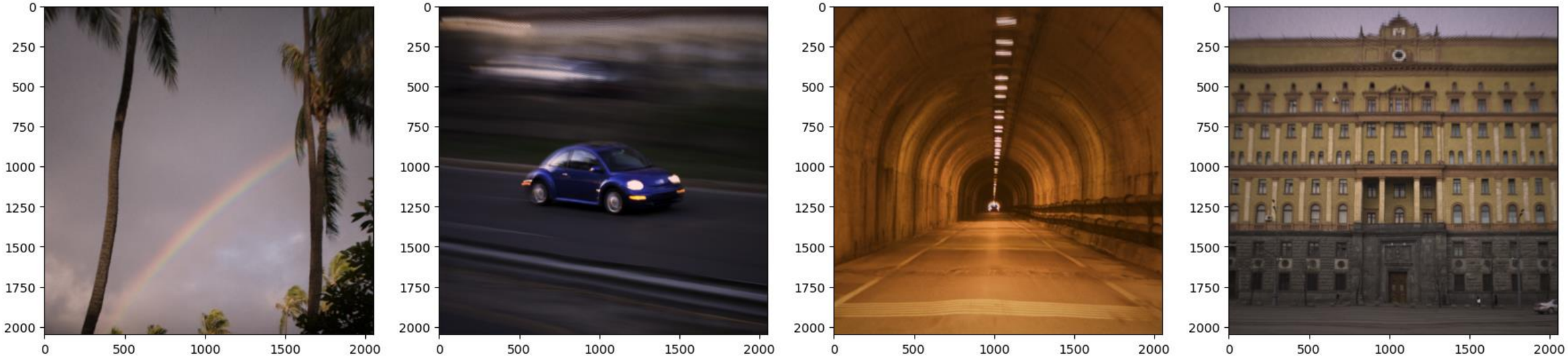




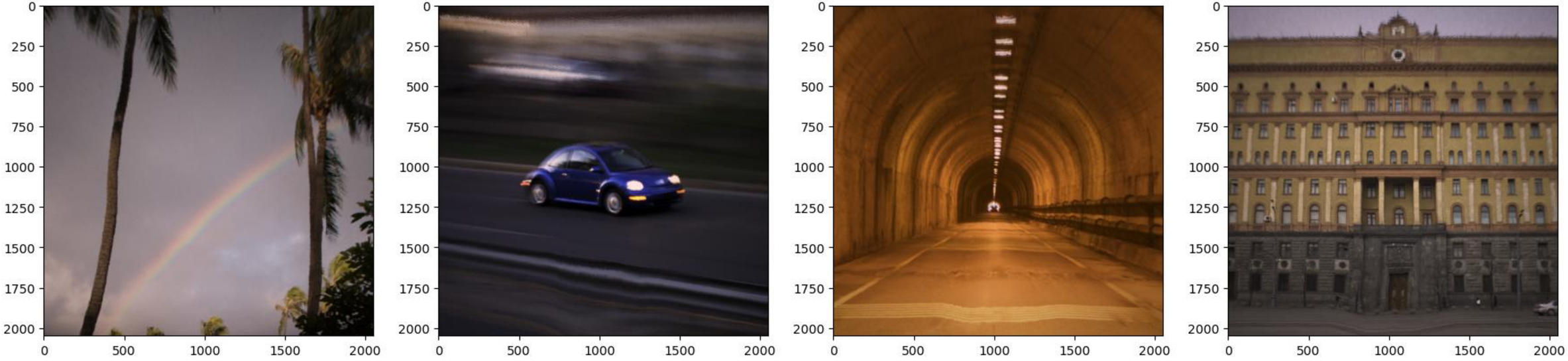
Reconstruction Using Dataset (at 256)



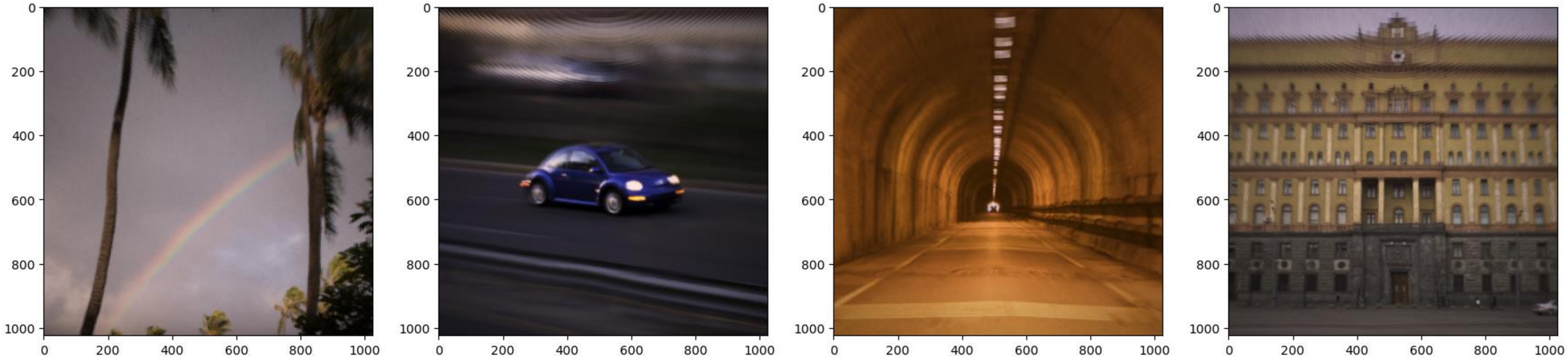
Resolution 2048 Reconstruction, Without Positional Encoding



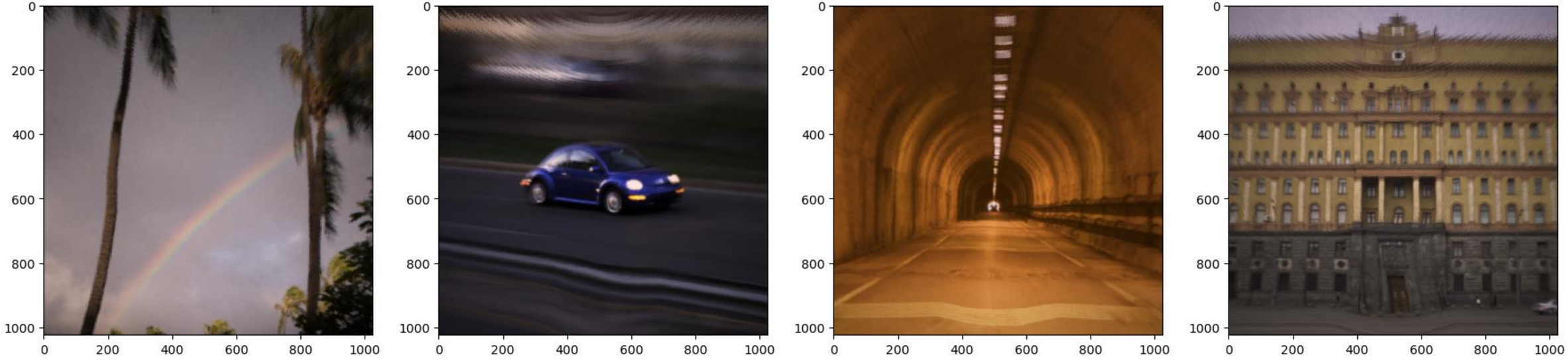
Resolution 2048 Reconstruction, With Positional Encoding



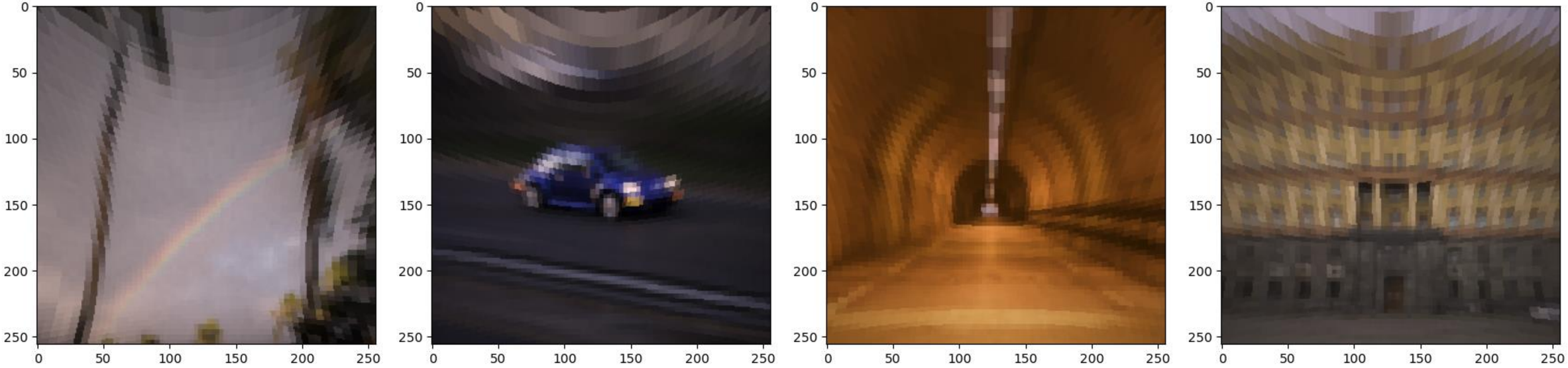
Resolution 1024 Reconstruction, Without Positional Encoding



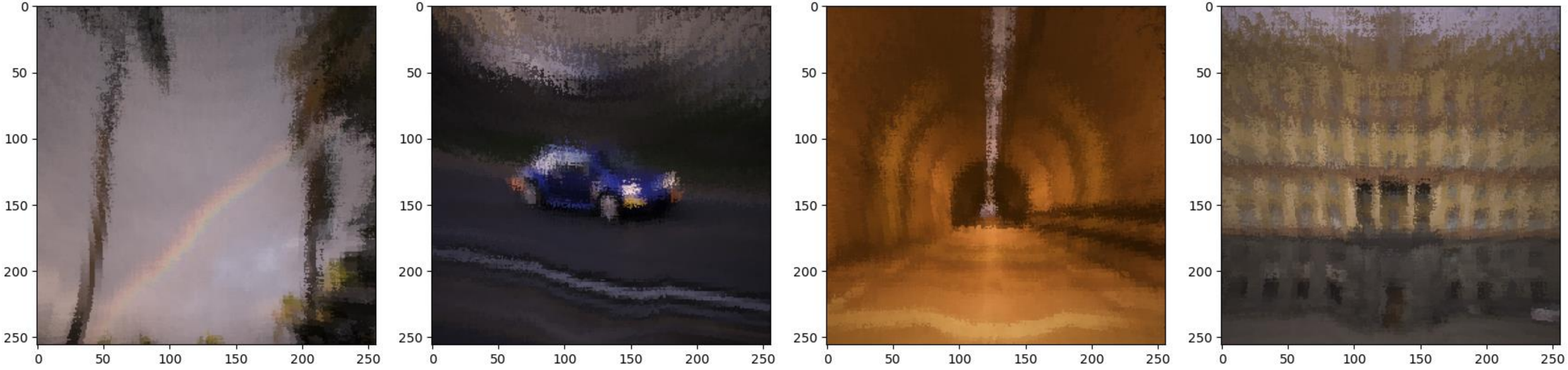
Resolution 1024 Reconstruction, With Positional Encoding



Resolution 256 Reconstruction, Without Positional Encoding



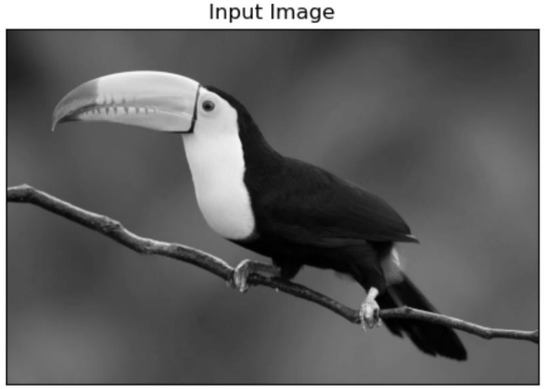
Resolution 256 Reconstruction, With Positional Encoding

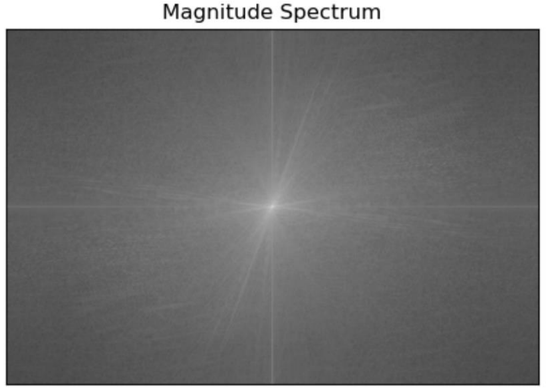


Compression analysis

Can we determine the required resolution to render at, depending on the totem geometry?

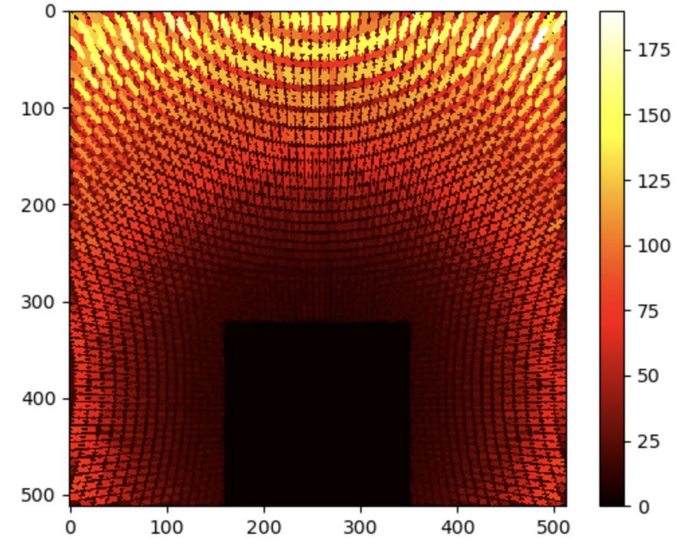
Need to spatially locate the various frequency components of the image

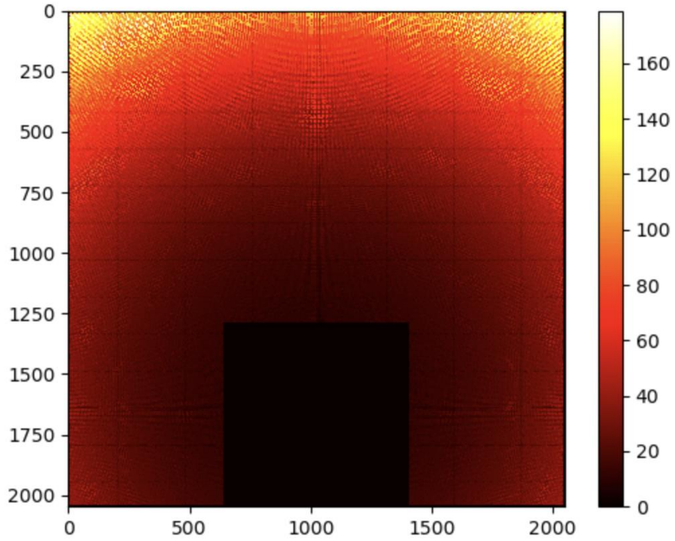




Fourier transform is insufficient because it convolves with a non localized sinusoid

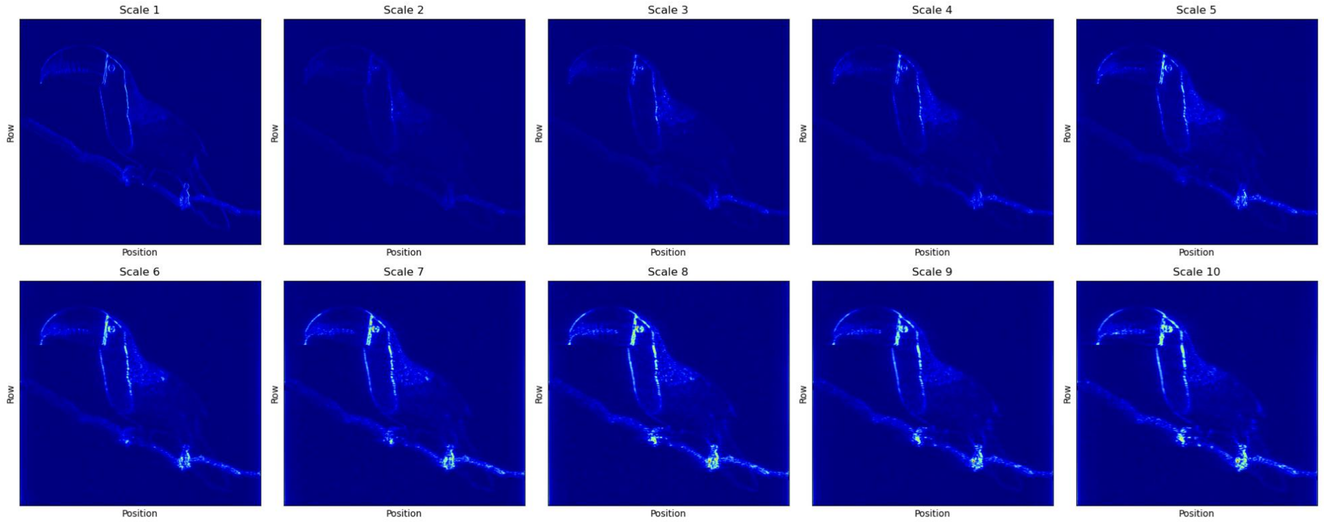
Shannon sampling theorem: $f_s >= 2f_{max}$





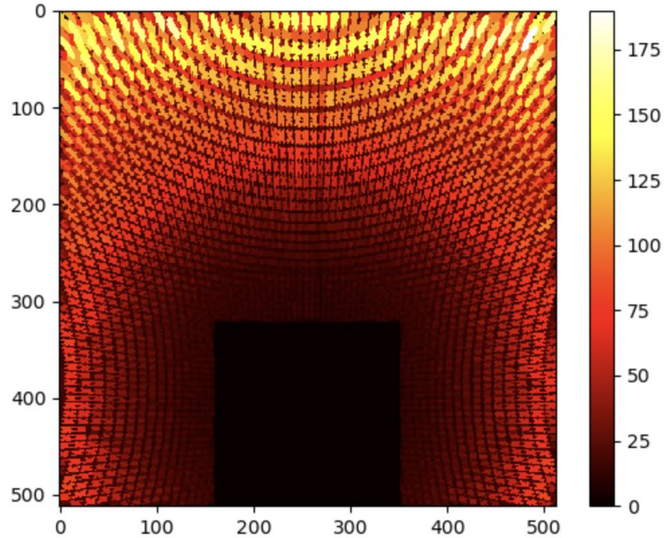
Compression heatmap $\propto 1/f_s$

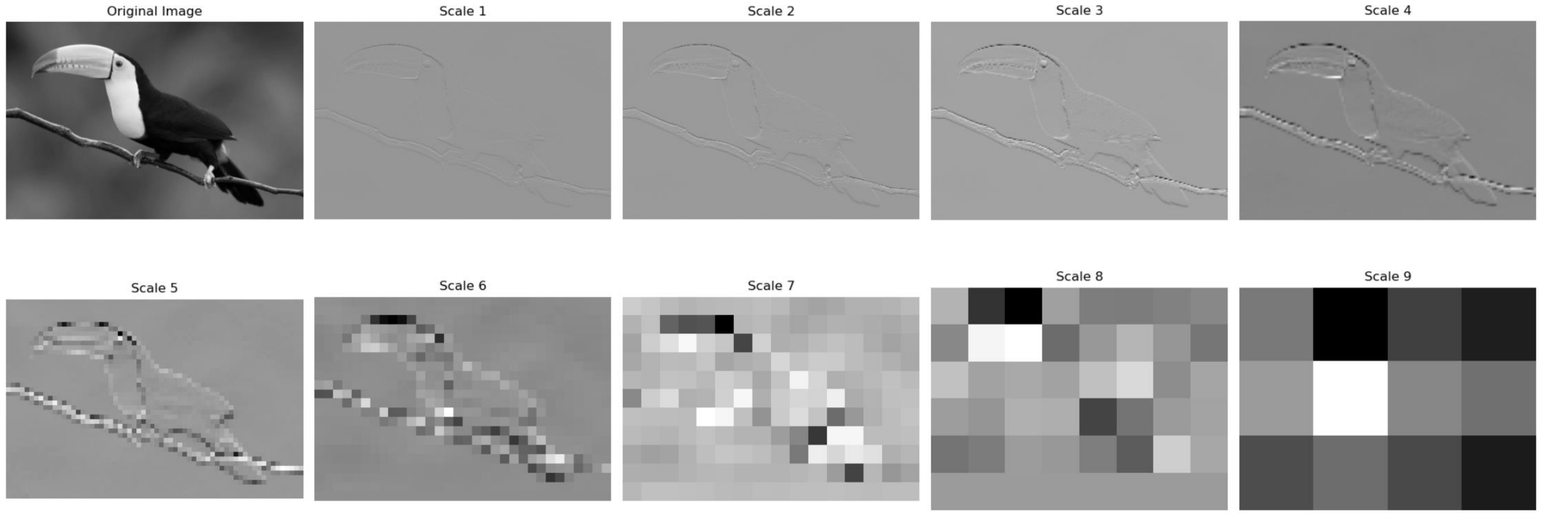
Compression analysis



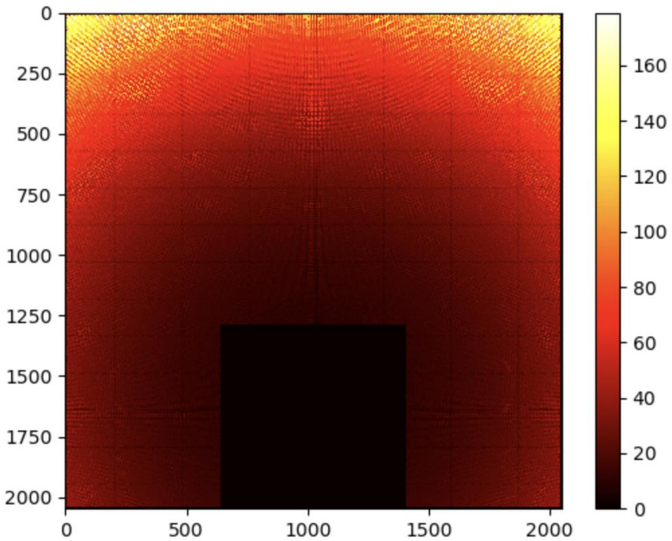
Shannon wavelet convolution

Shannon sampling theorem: $f_s >= 2f_{max}$



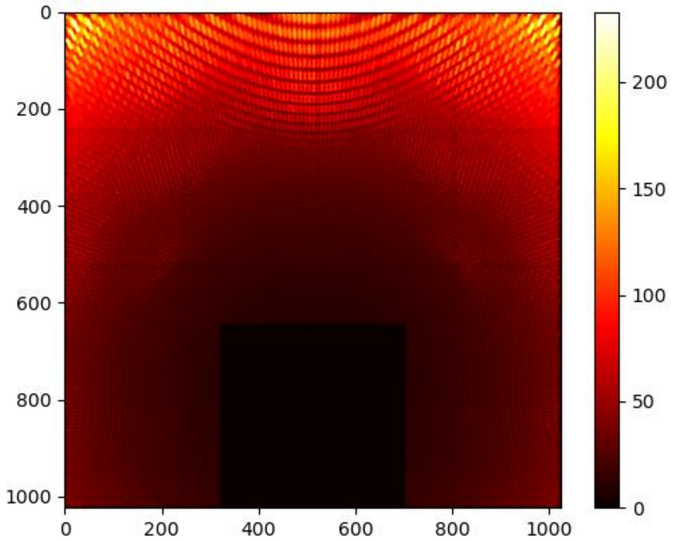


Haar wavelet convolution

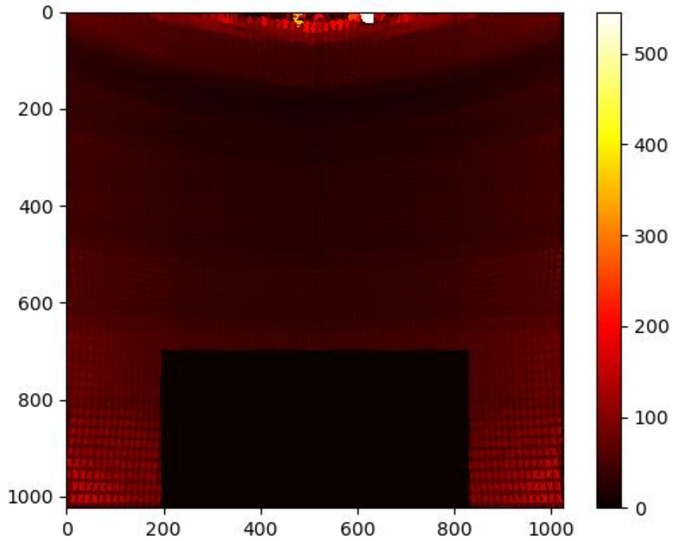


Compression heatmap $\propto 1/f_s$

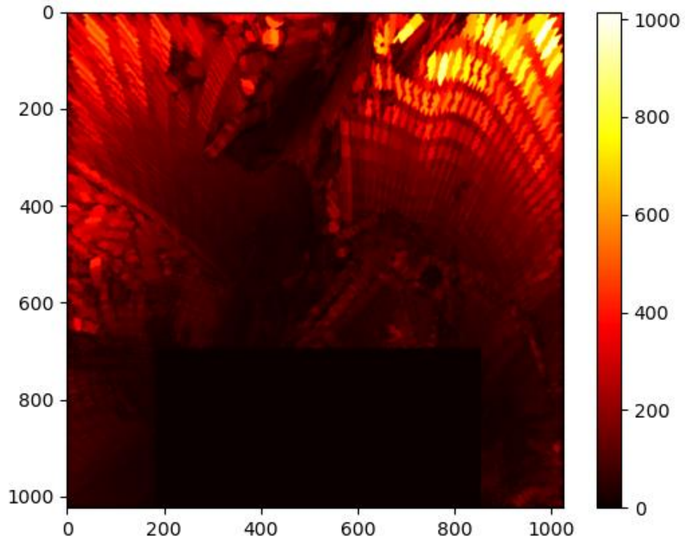
Compression heatmaps



Sphere

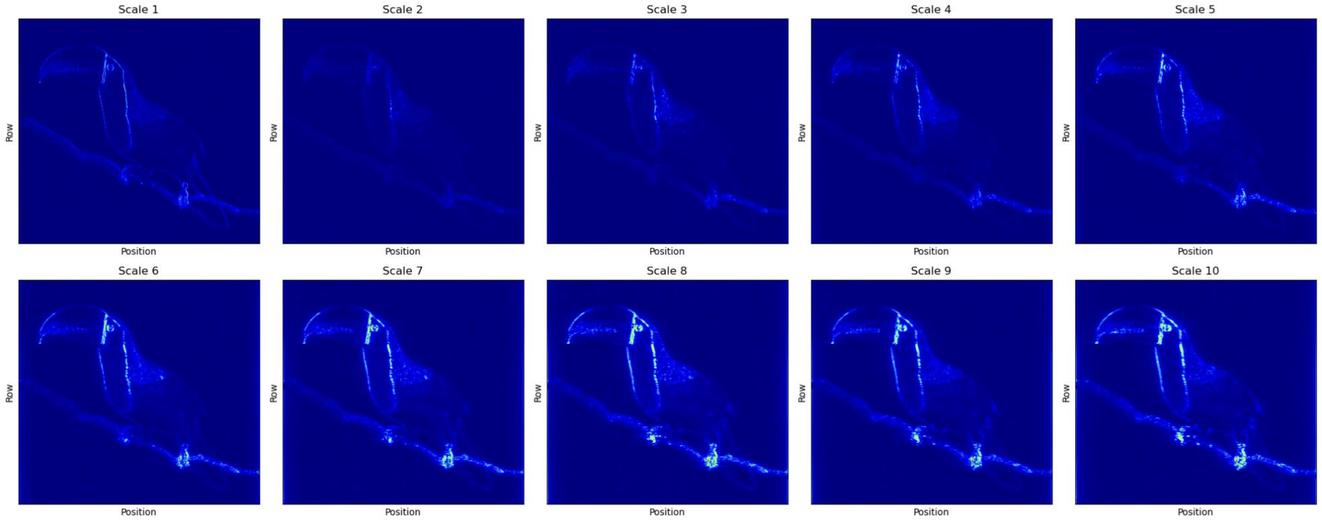


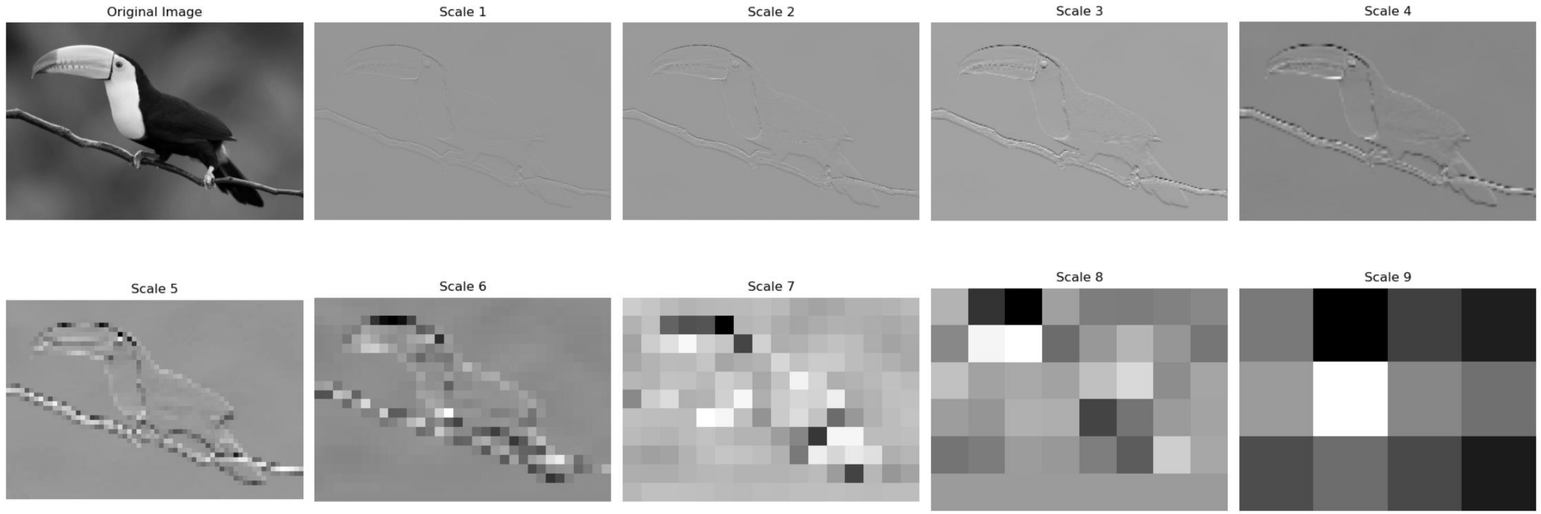
Teapot

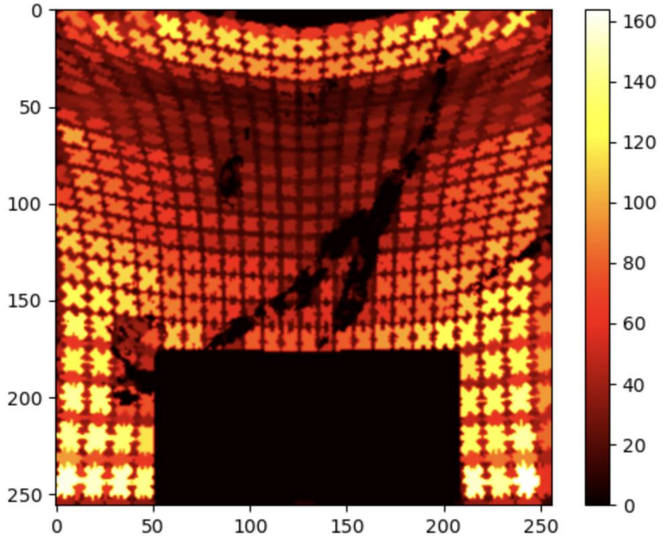


Knot

Compression analysis for teapot

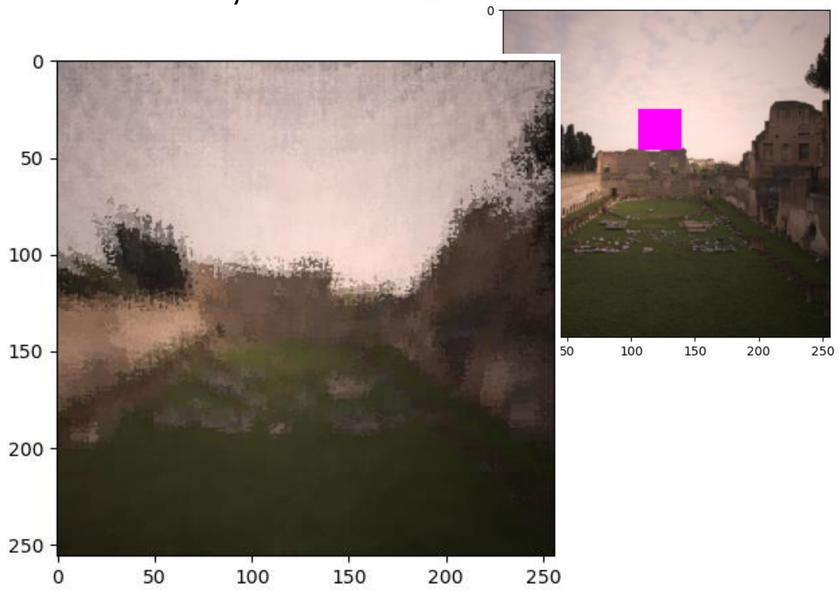


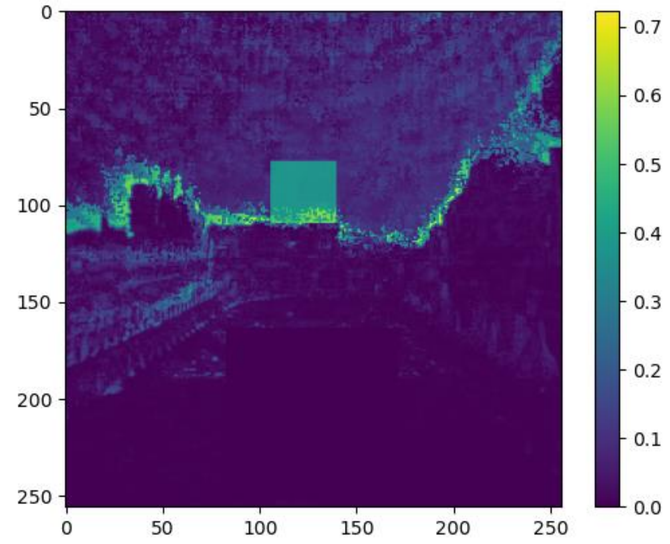




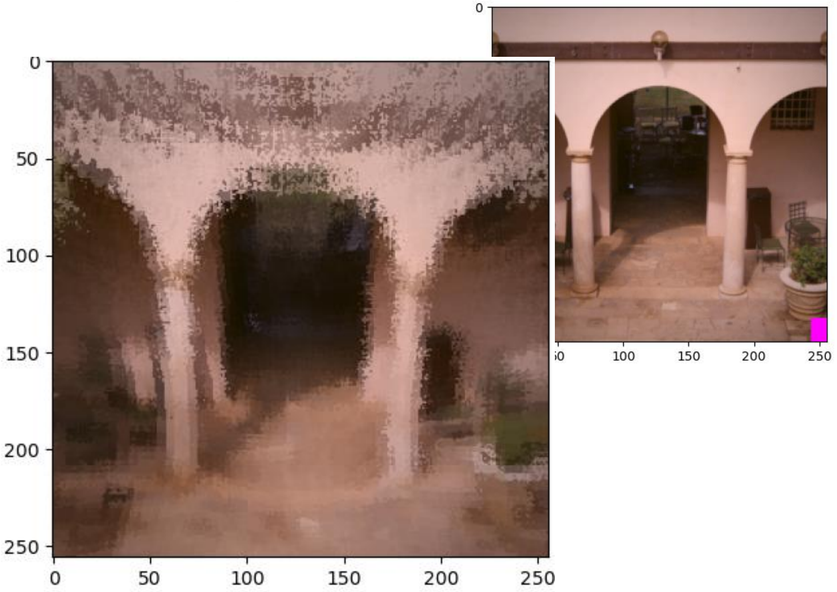
Examples of outliers

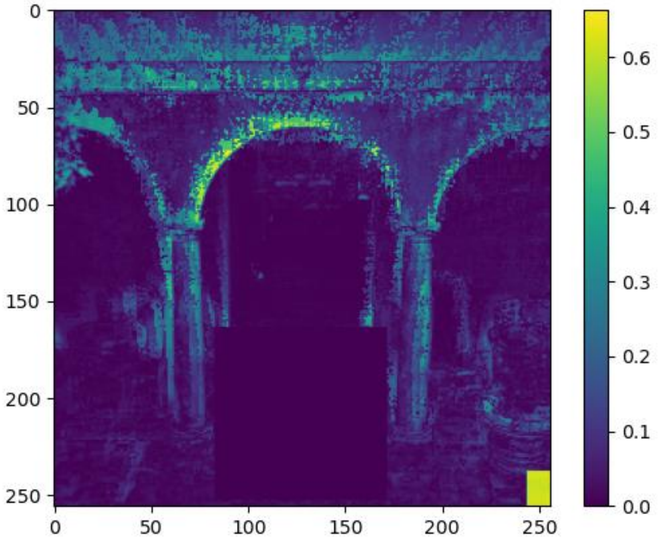
AP: 0.37; PSNR: 19.04



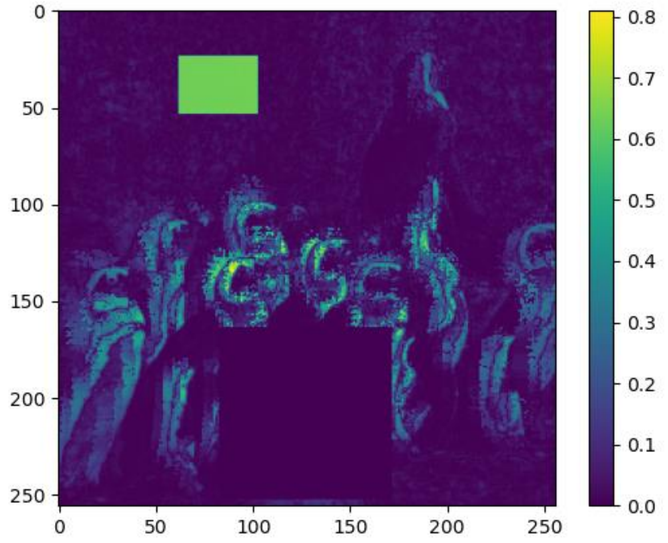


AP: 0.76; PSNR: 17.07

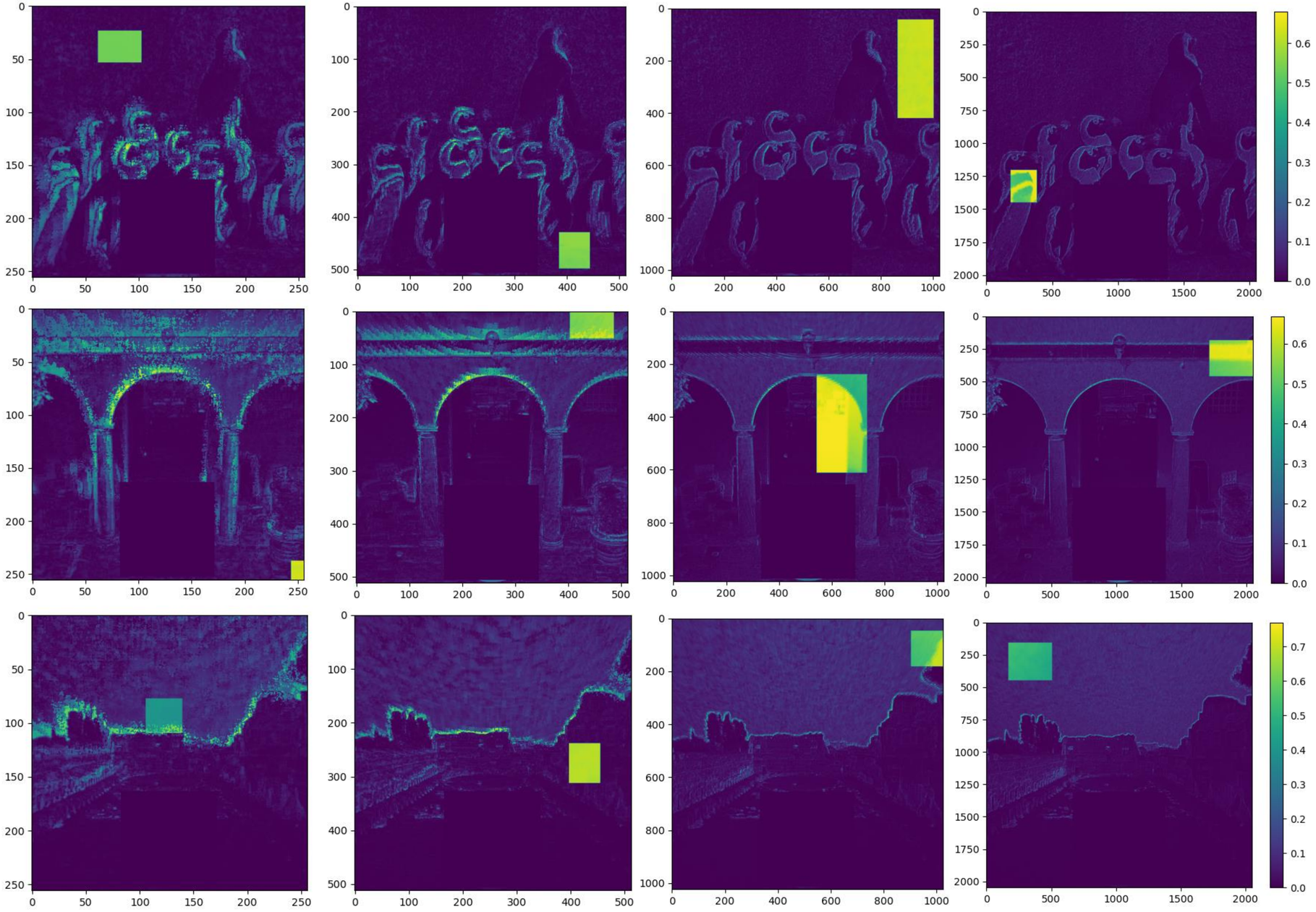




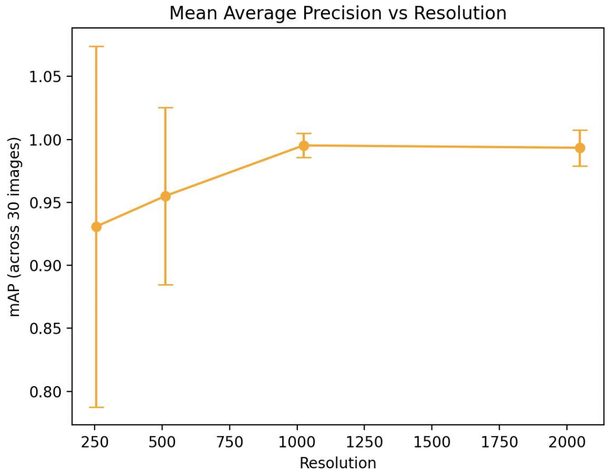
AP: 0.83; PSNR: 17.51

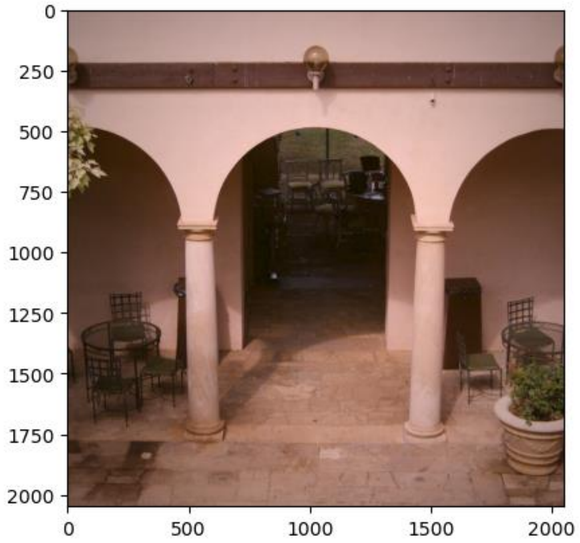


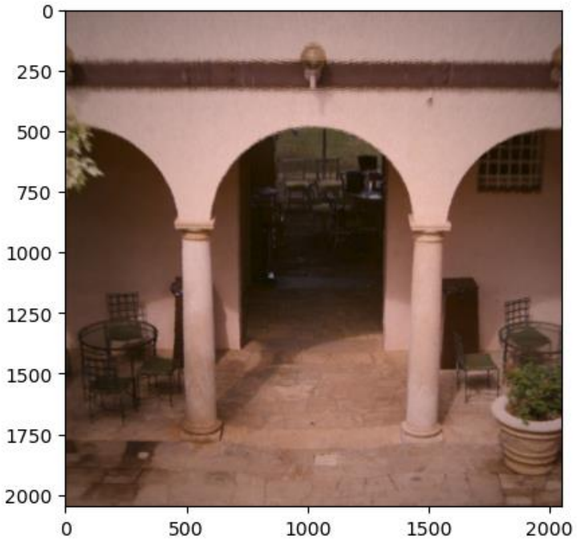
How do the weaker results change with increasing resolution?

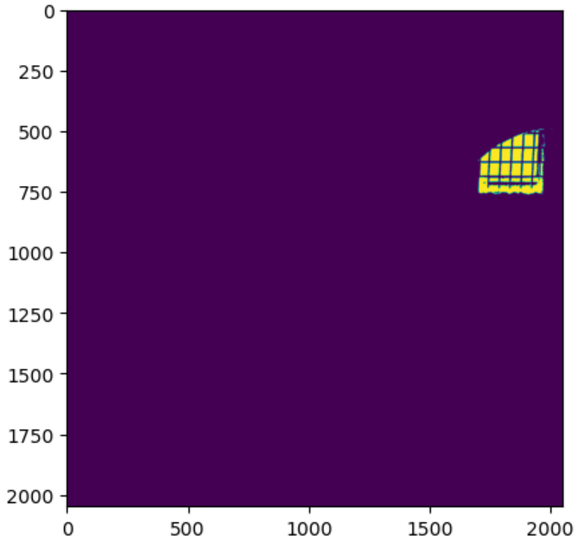


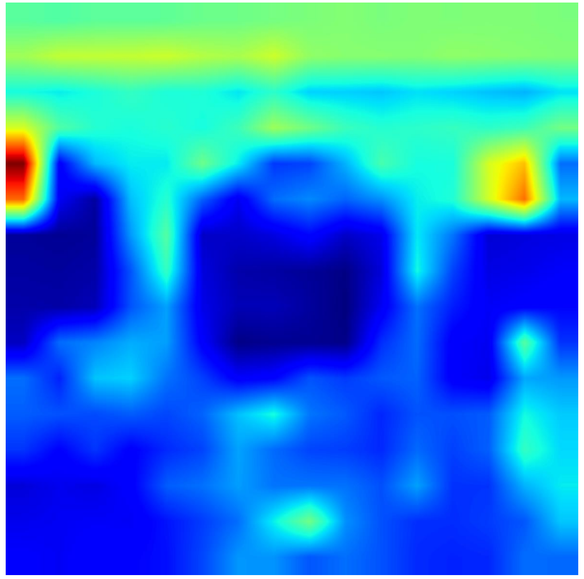
Res 256 512 1024 2048







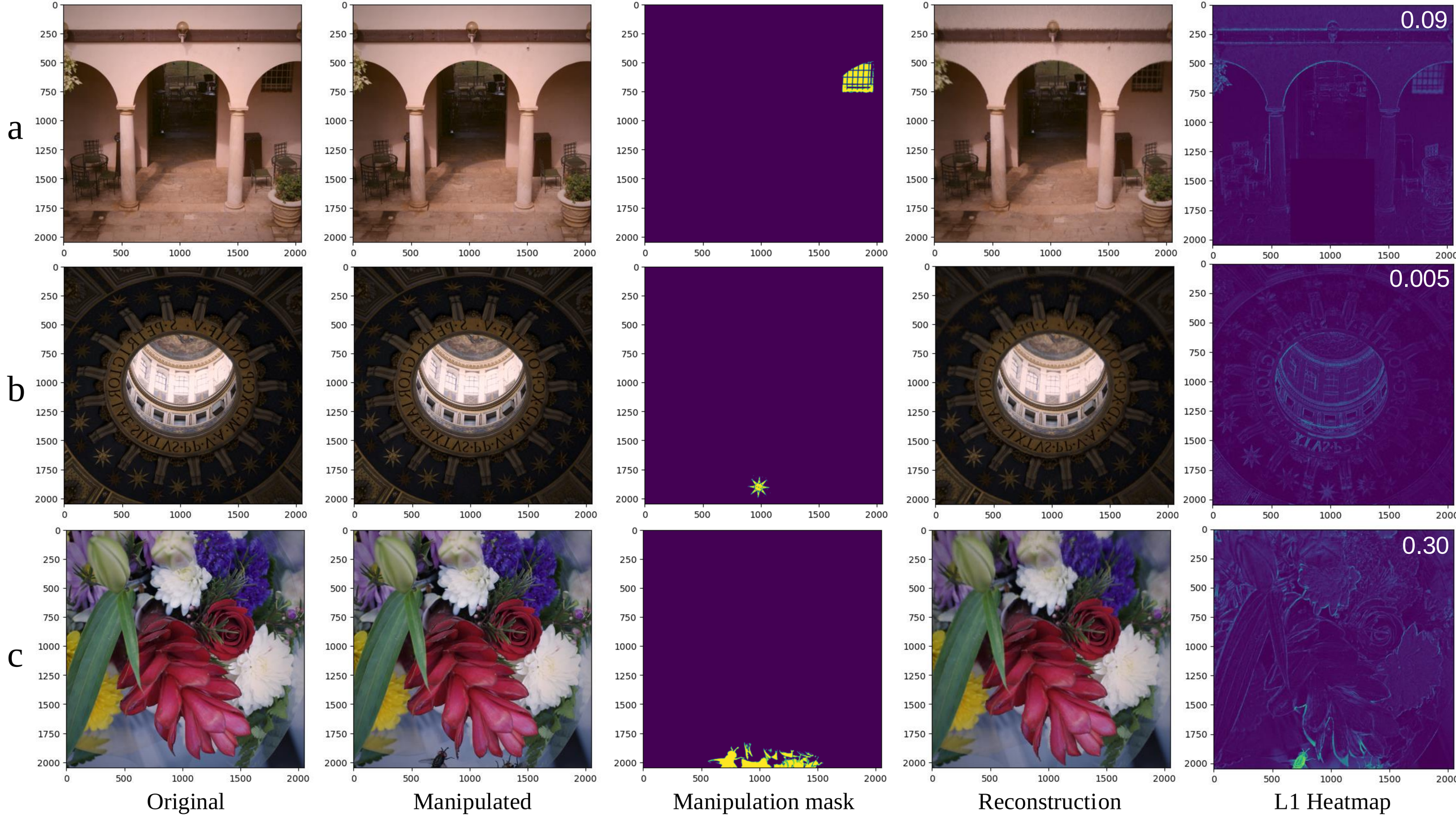


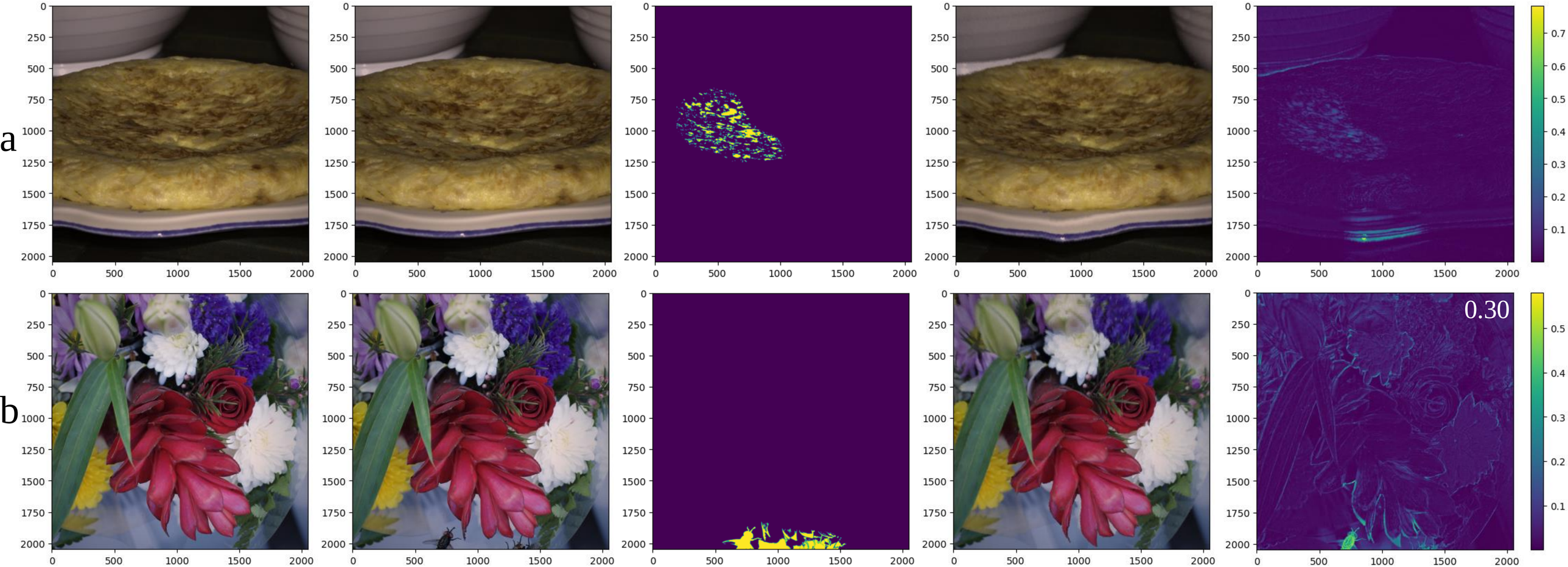


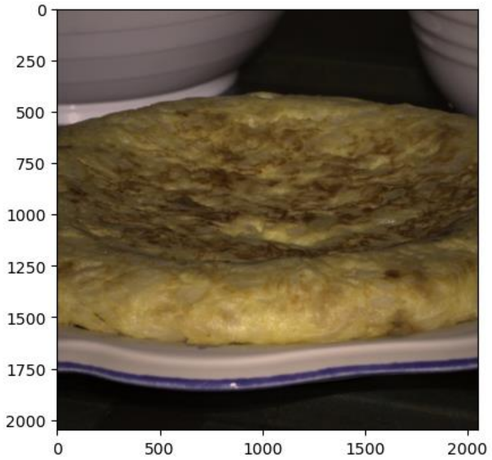
0.1693655496833258

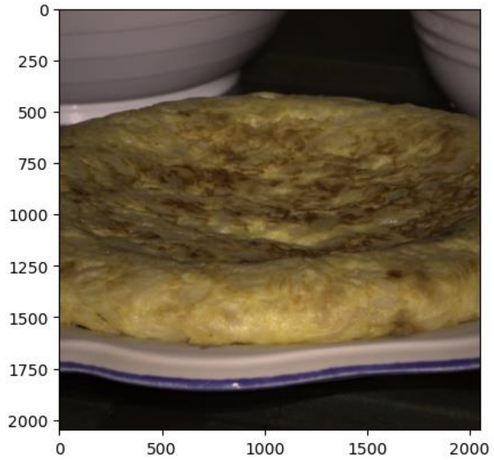
0.10581424740582294

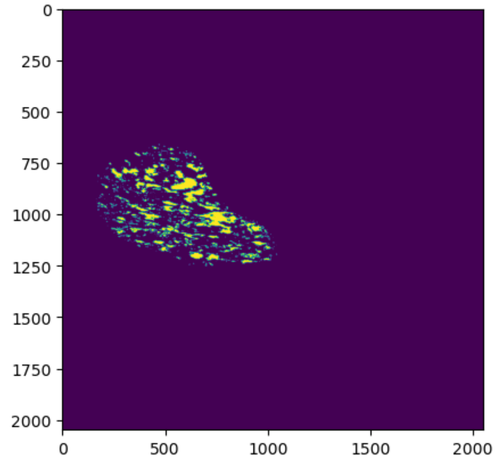
0.07974349091330014

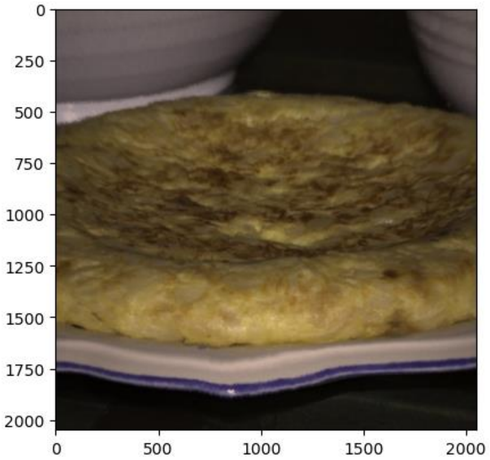

a
b
c
0.09
0.005
0.30
Original
Manipulated
Manipulation mask
Reconstruction
L1 Heatmap

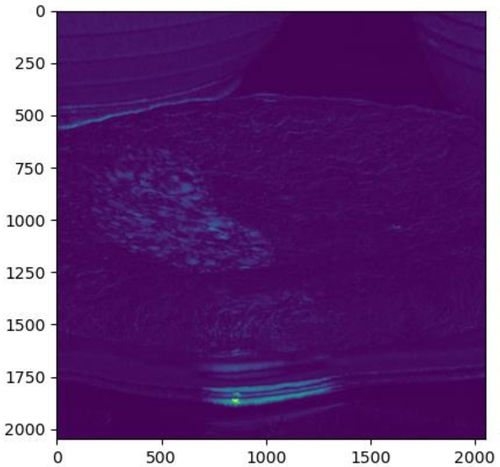

a
b
0.30
Original
Manipulated
Manipulation mask
Reconstruction
Inconsistency Heatmap

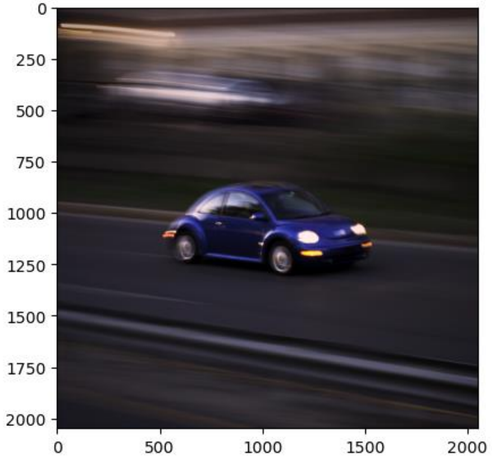

Original

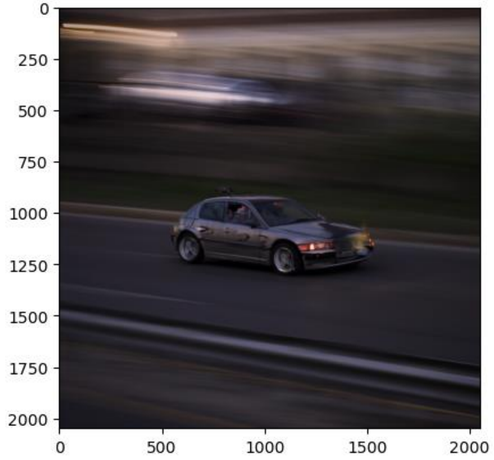

Manipulated

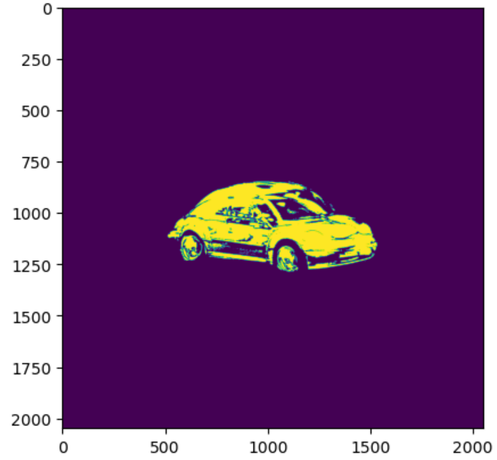

Manipulation mask

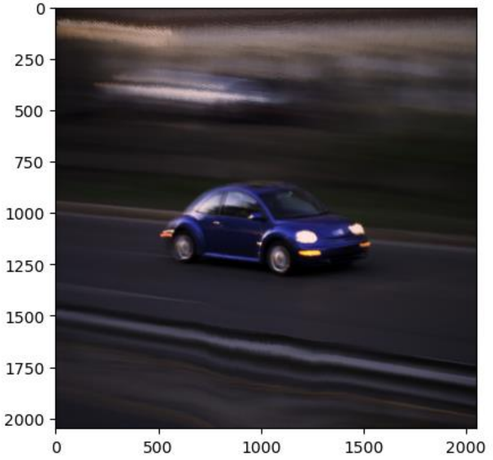

Reconstruction

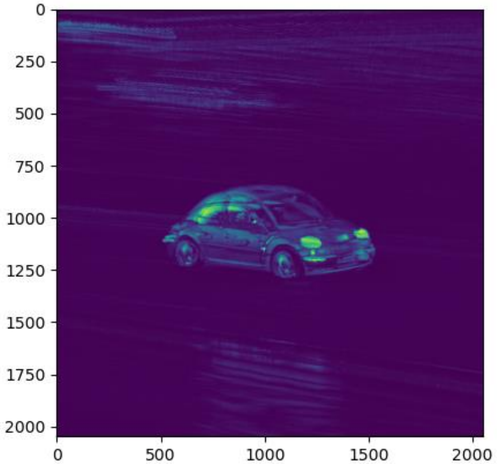

L1 Heatmap
AP 0.26

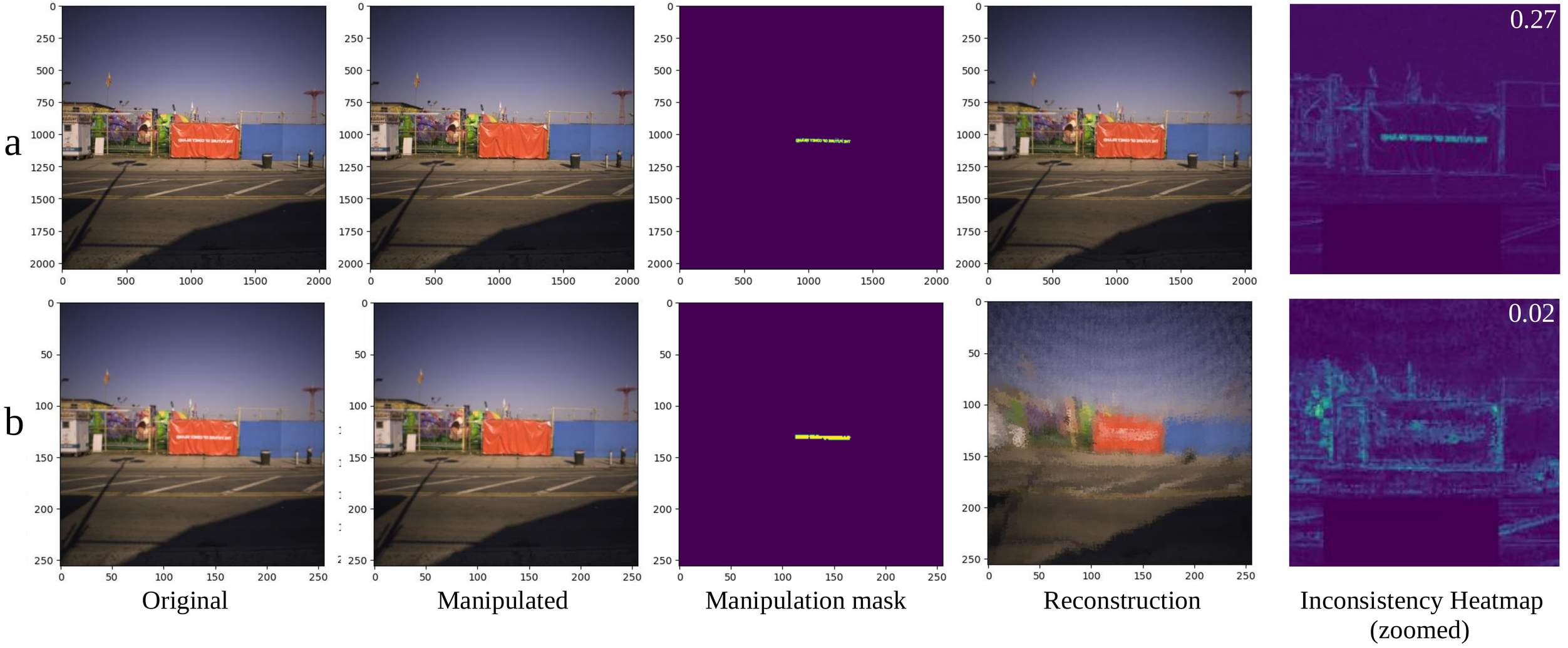

Original


Manipulated

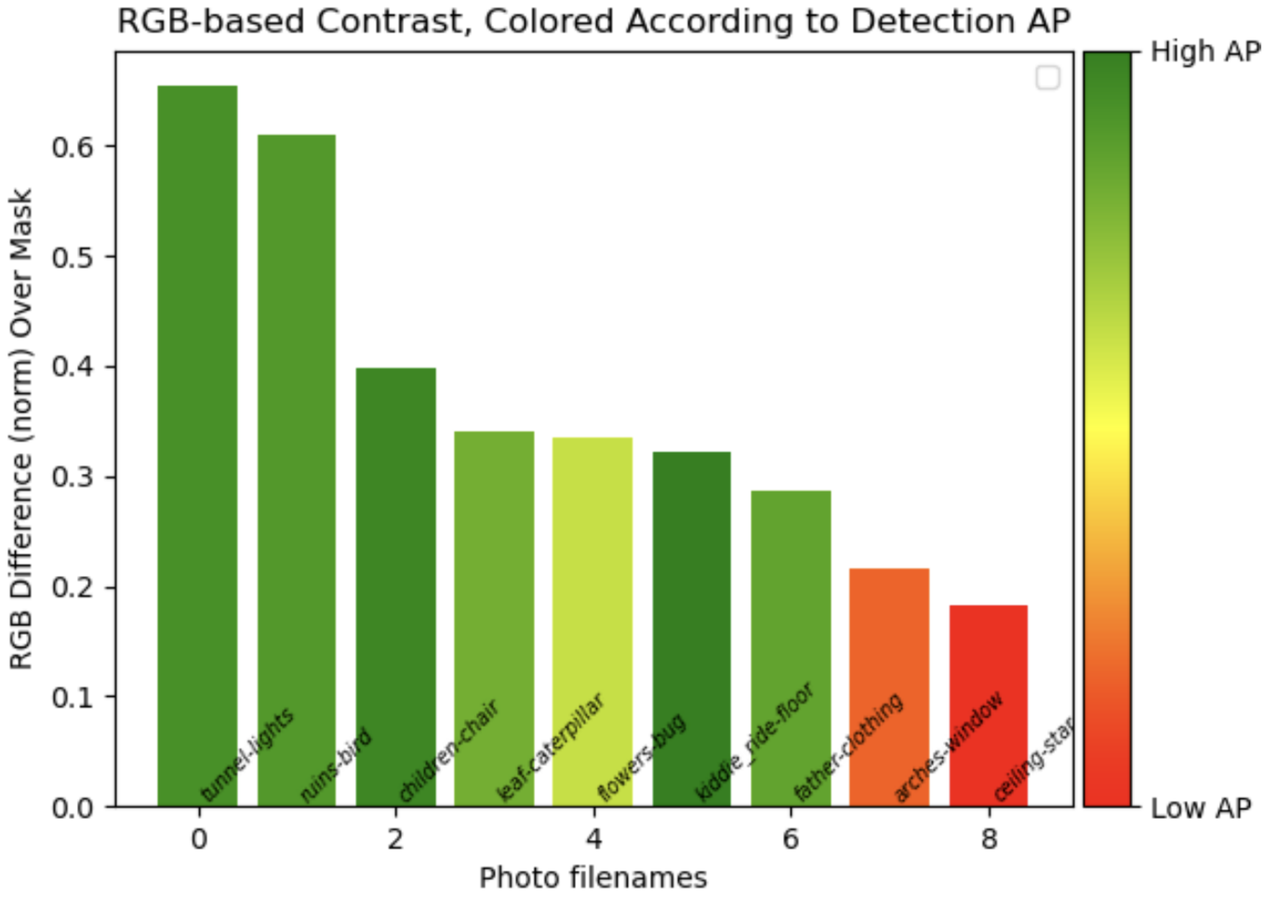

Manipulation mask


Reconstruction

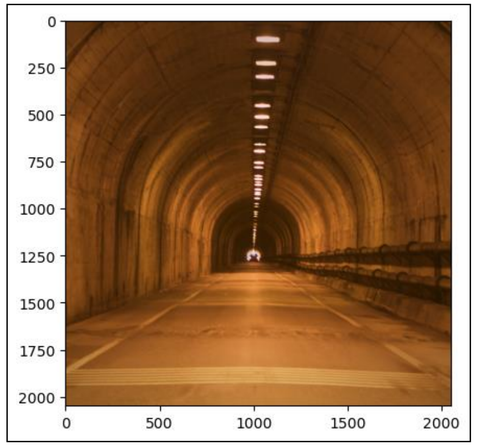

L1 Heatmap
AP 0.79


a
b
0.27
0.02
Original
Manipulated
Manipulation mask
Reconstruction
Inconsistency Heatmap
(zoomed)

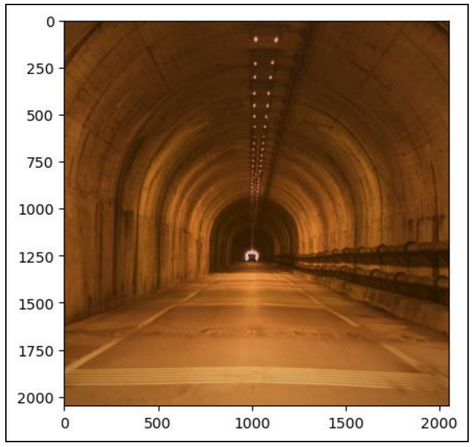
Contrast-AP correlation study



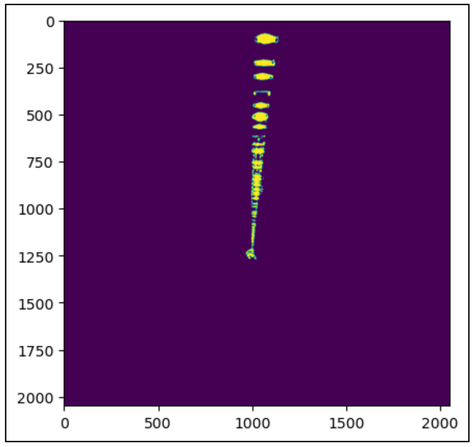
A first step at verifying the correlation between high contrast and high AP



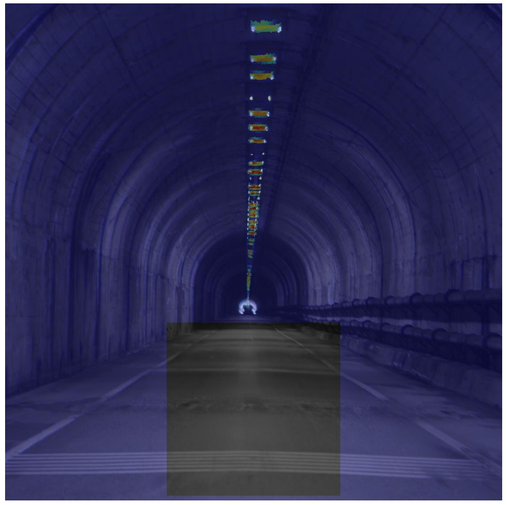
Original



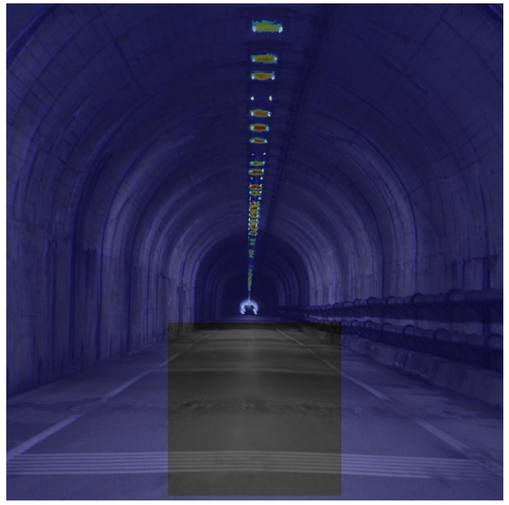
Manipulated



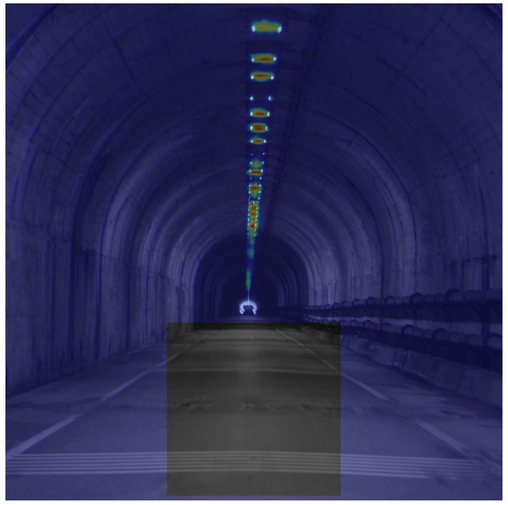
Manipulation Mask



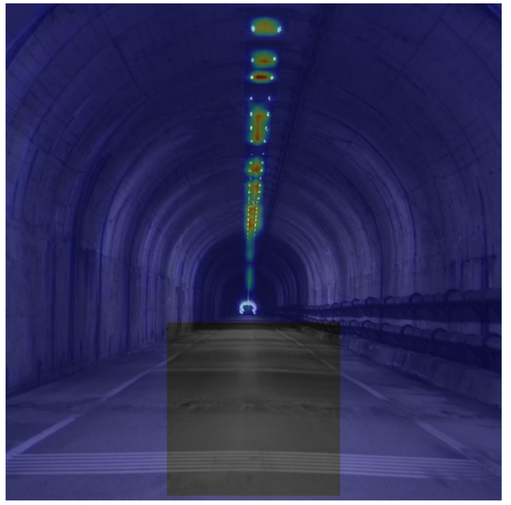
patch size 1 AP 0.82



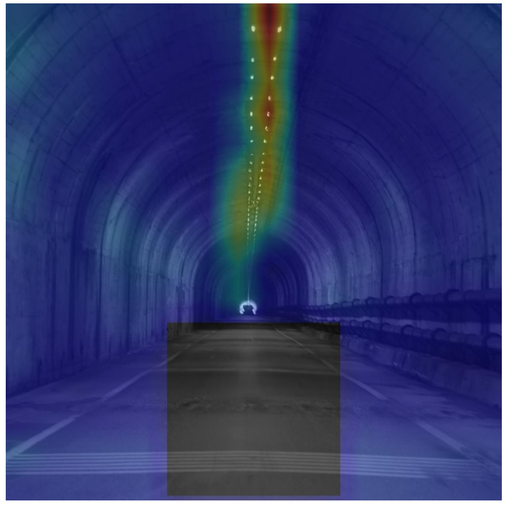
patch size 4 AP 0.89



patch size 16 AP 0.88

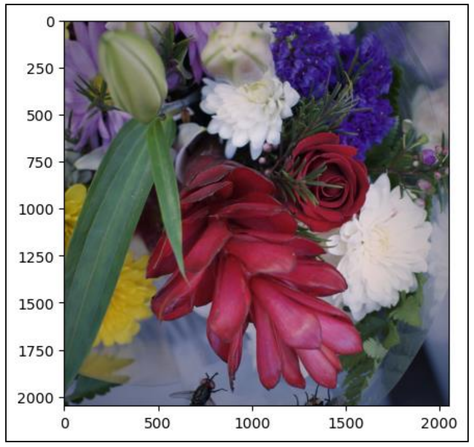


patch size 32 AP 0.74

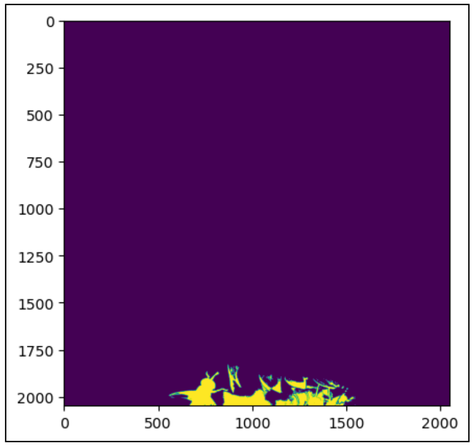


patch size 128 AP 0.22

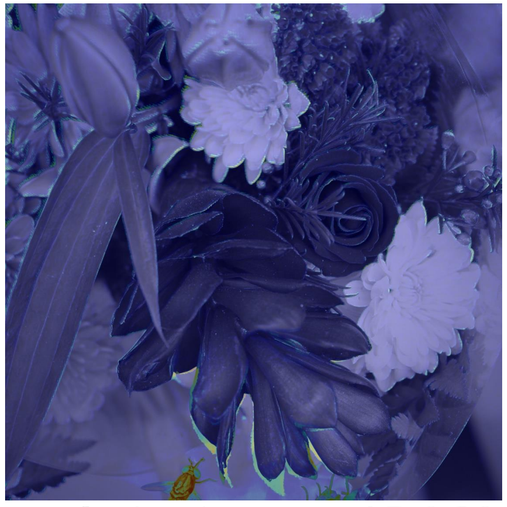
Original



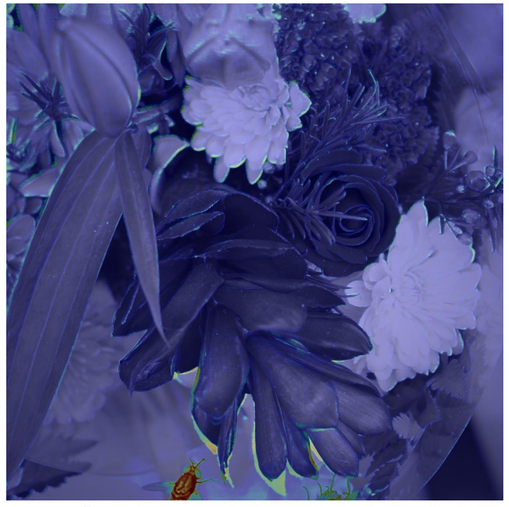
Manipulated



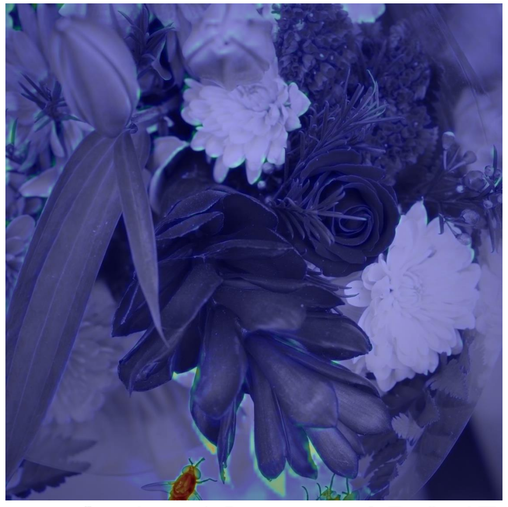
Manipulation Mask



patch size 1 AP 0.30

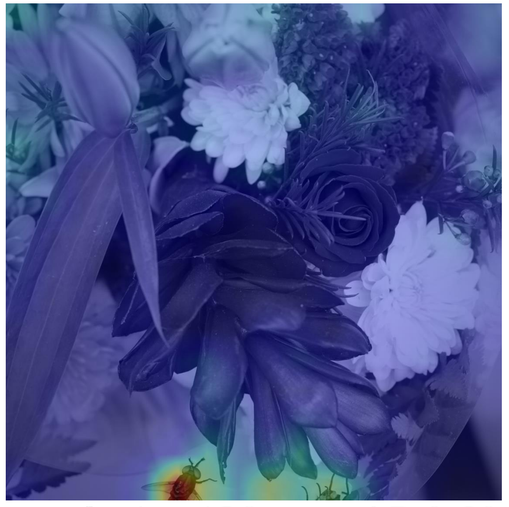


patch size 4 AP 0.37

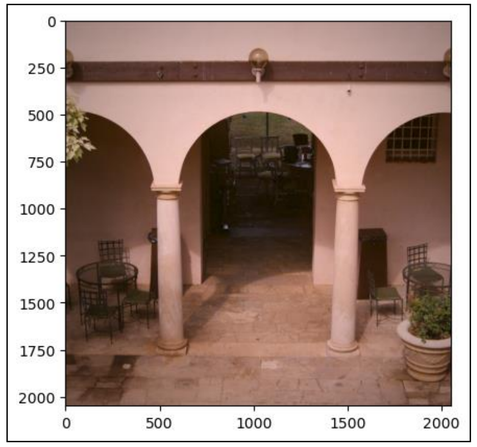


patch size 16 AP 0.47

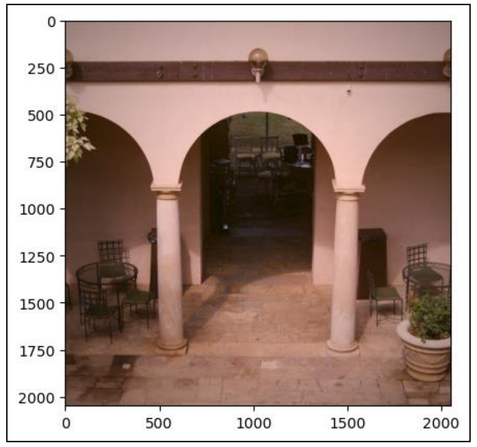
patch size 32 AP 0.55



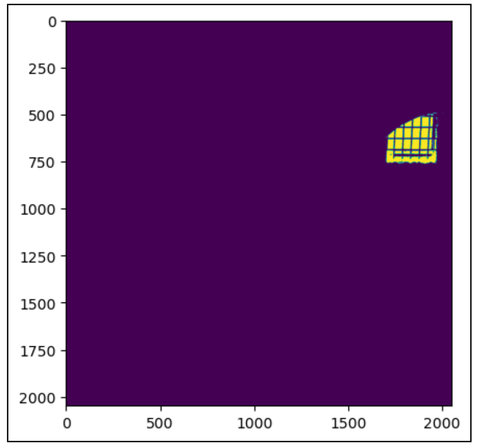
patch size 128 AP 0.60



Original



Manipulated



Manipulation Mask

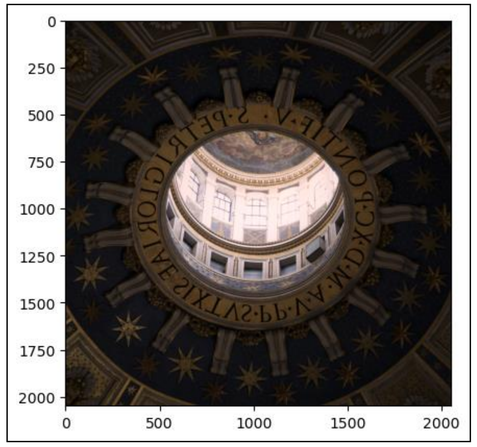
patch size 1 AP 0.09

patch size 4 AP 0.10

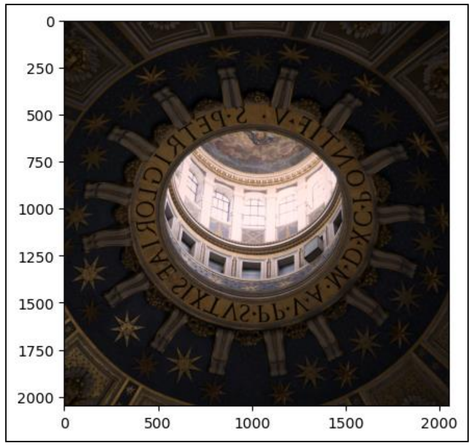
patch size 16 AP 0.11

patch size 32 AP 0.12

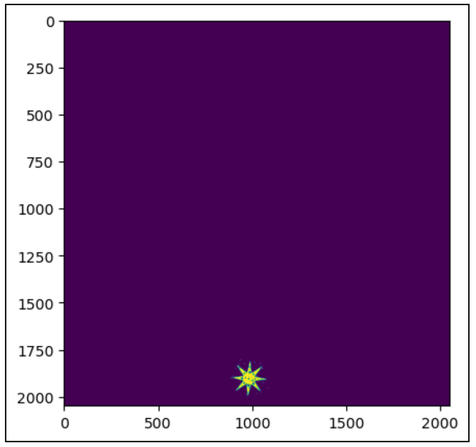
patch size 128 AP 0.17



Original



Manipulated



Manipulation Mask

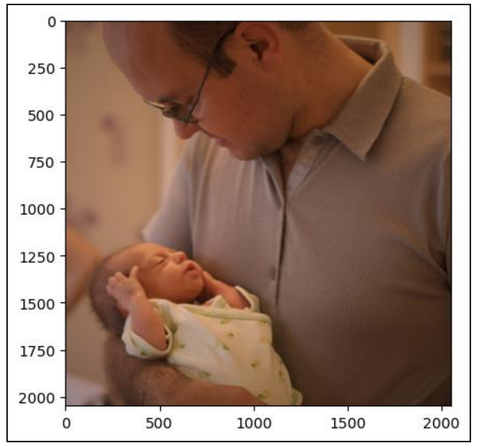
patch size 1 AP 0.0046

patch size 4 AP 0.0049

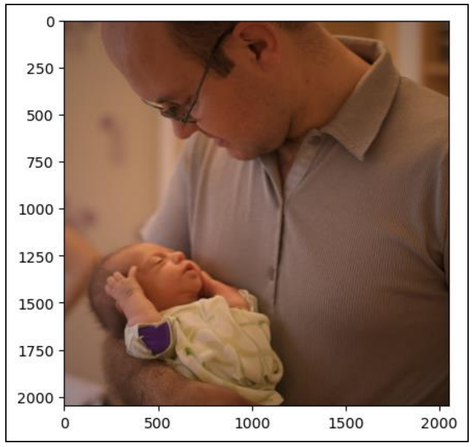
patch size 16 AP 0.0050

patch size 32 AP 0.0048

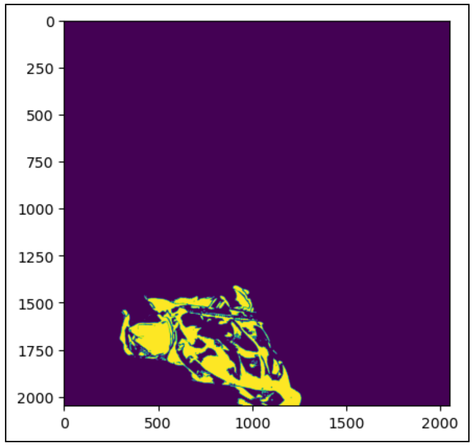
patch size 128 AP 0.0031



Original



Manipulated



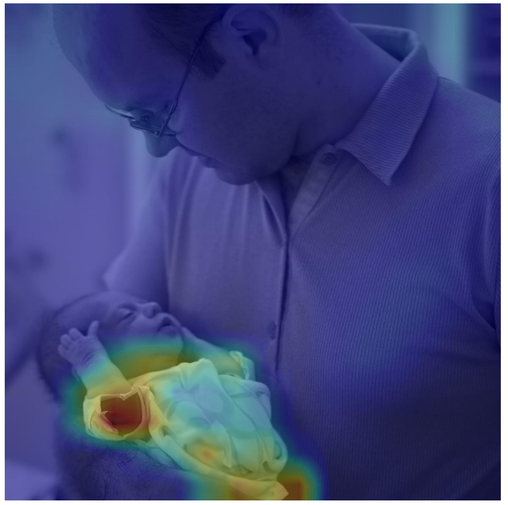
Manipulation Mask

patch size 1 AP 0.76

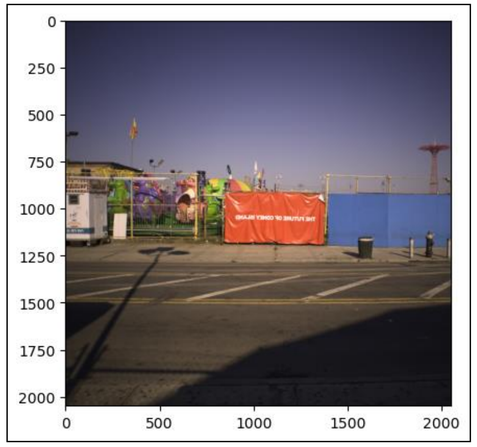
patch size 4 AP 0.80

patch size 16 AP 0.82

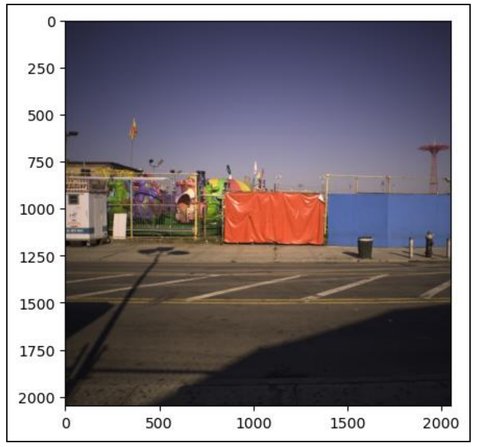
patch size 32 AP 0.80



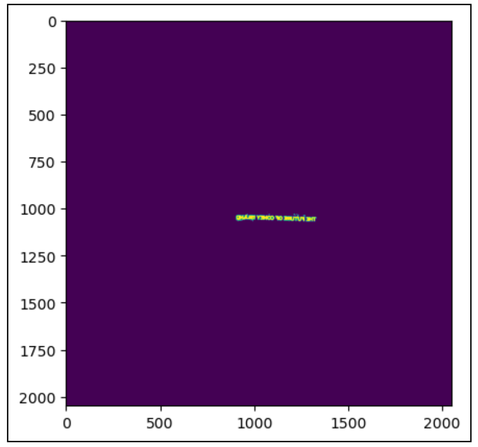
patch size 128 AP 0.66



Original



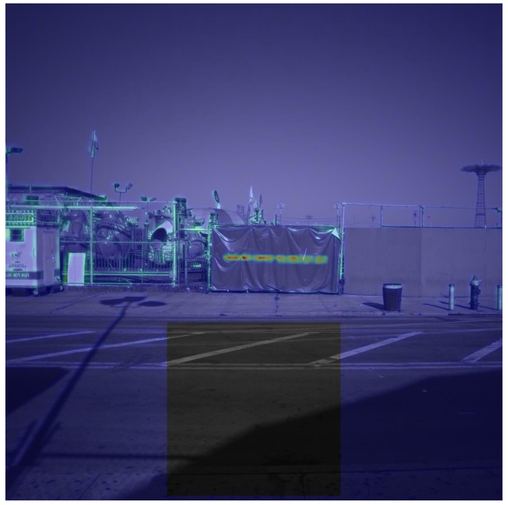
Manipulated



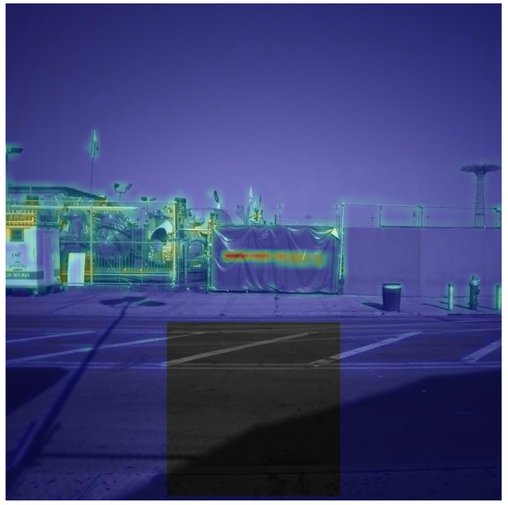
Manipulation Mask

patch size 1 AP 0.35

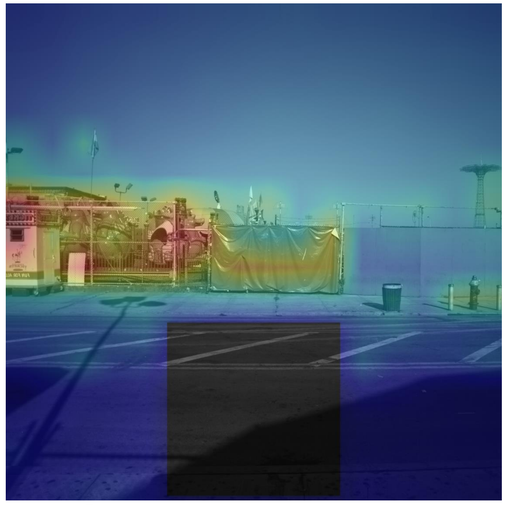
patch size 4 AP 0.64



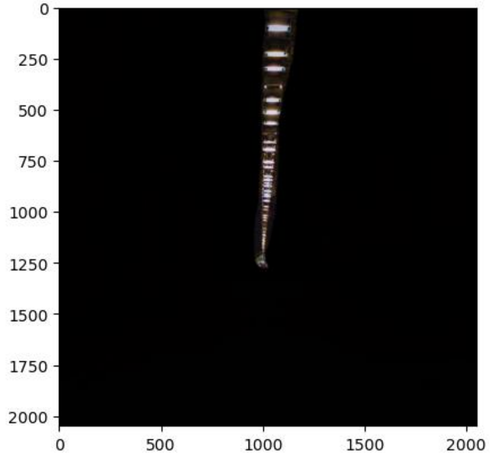
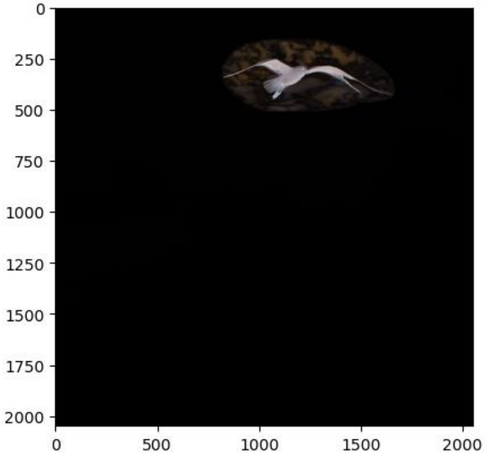
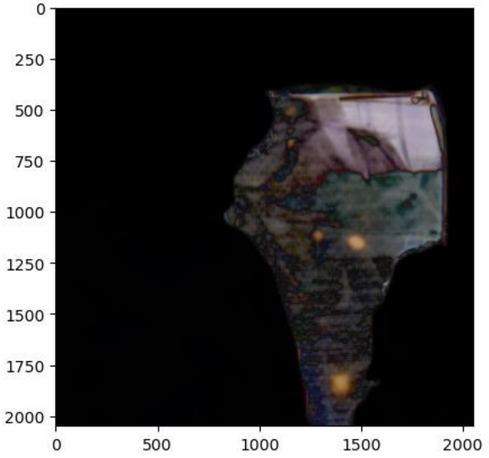
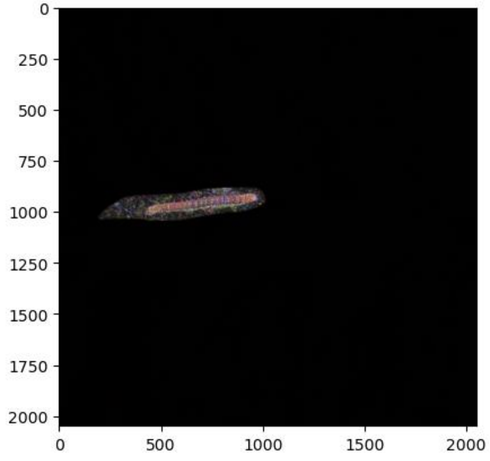
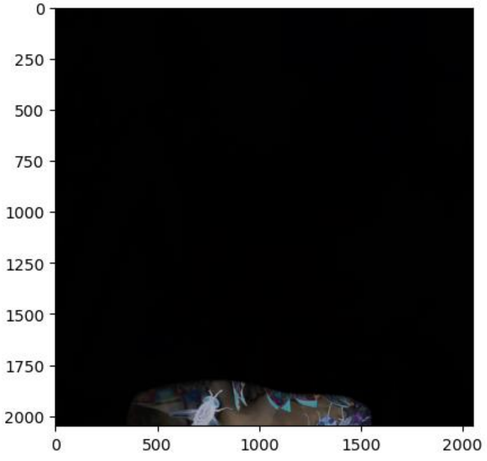
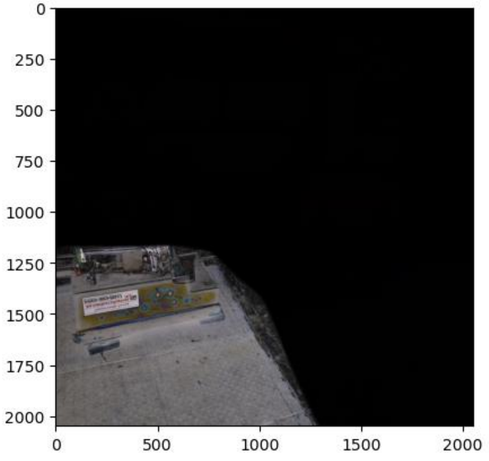
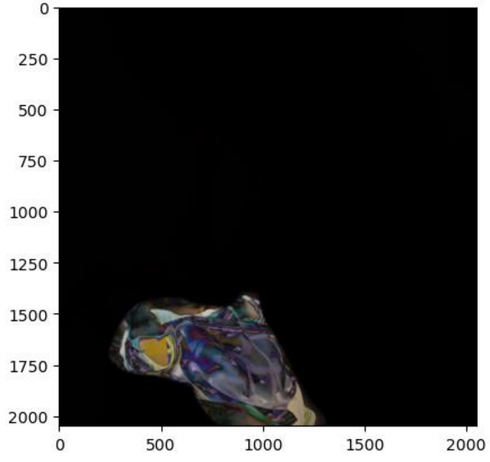
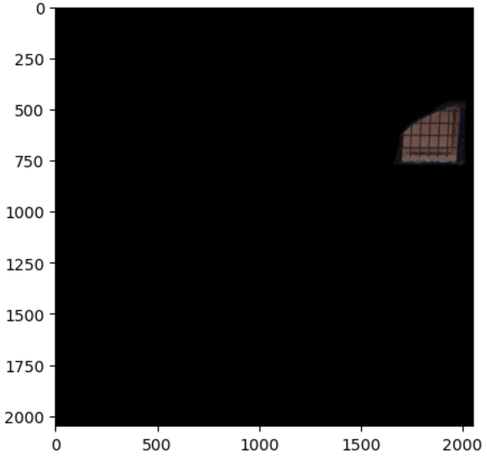
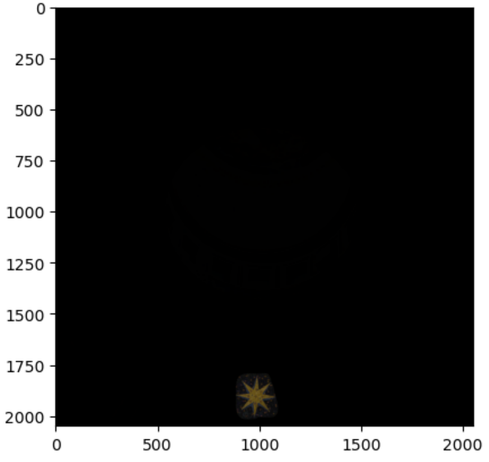
patch size 16 AP 0.78



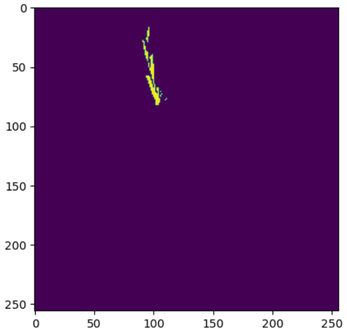
patch size 32 AP 0.31



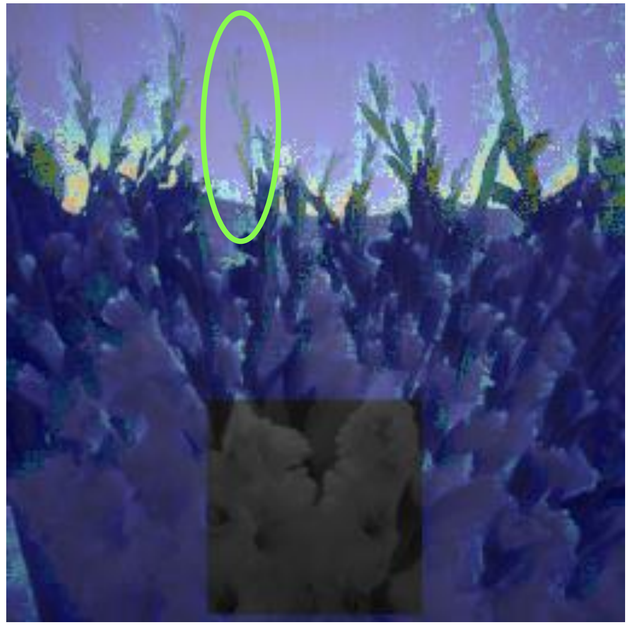
patch size 128 AP 0.012

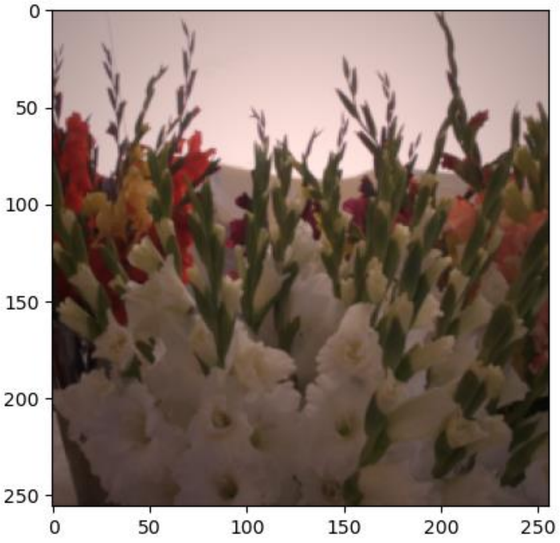










top mask



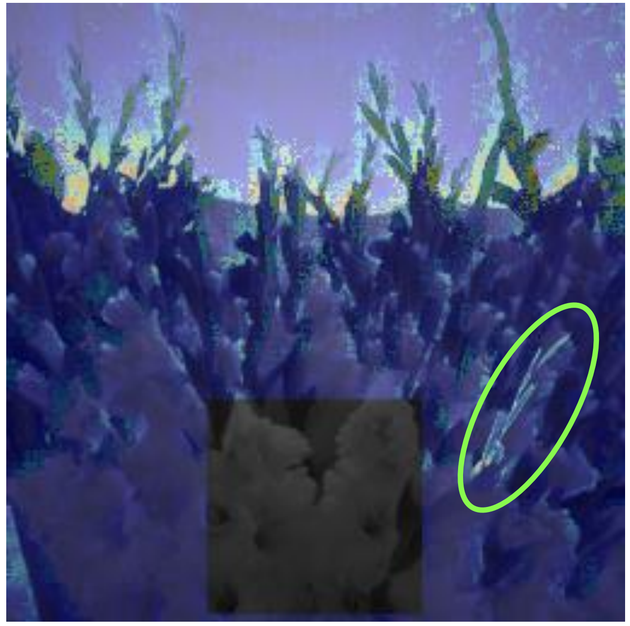
“top” manipulation



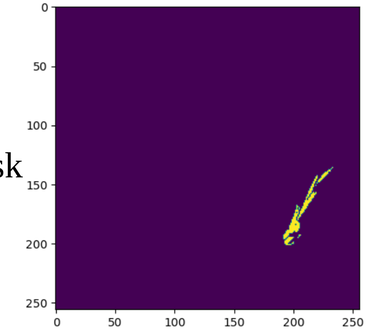


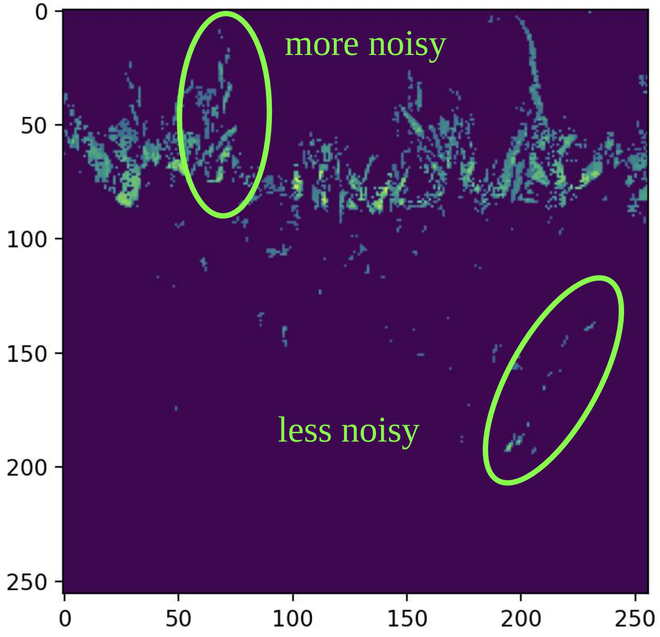
Original

“bottom” manipulation



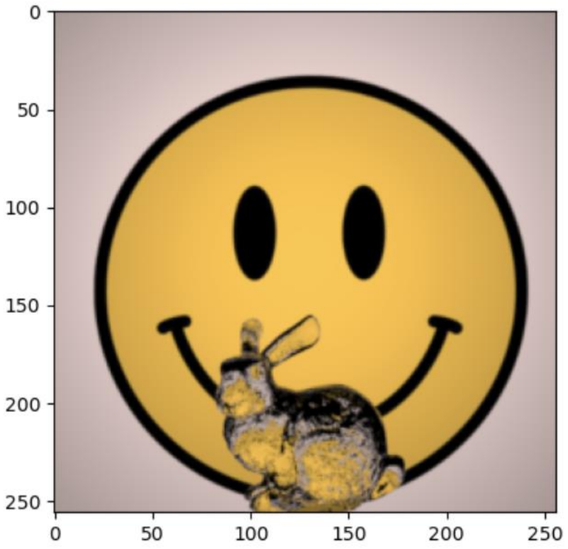
bottom mask

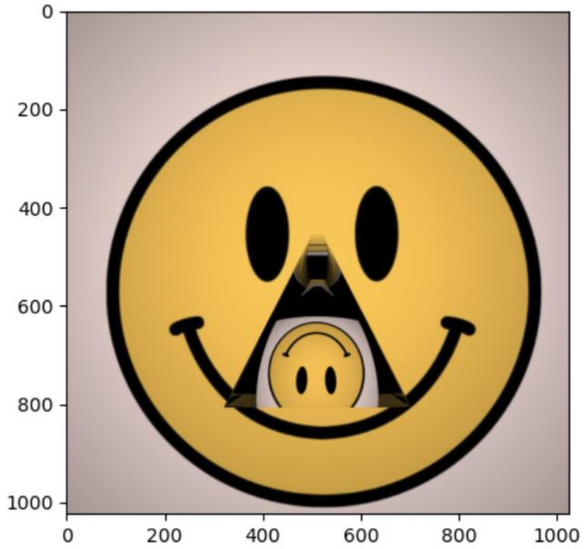


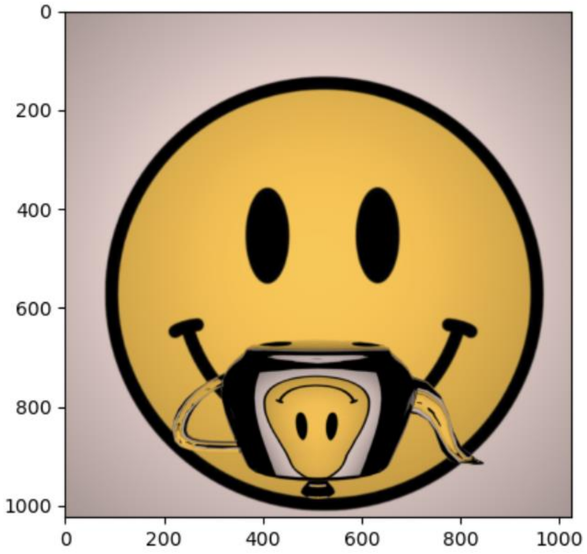


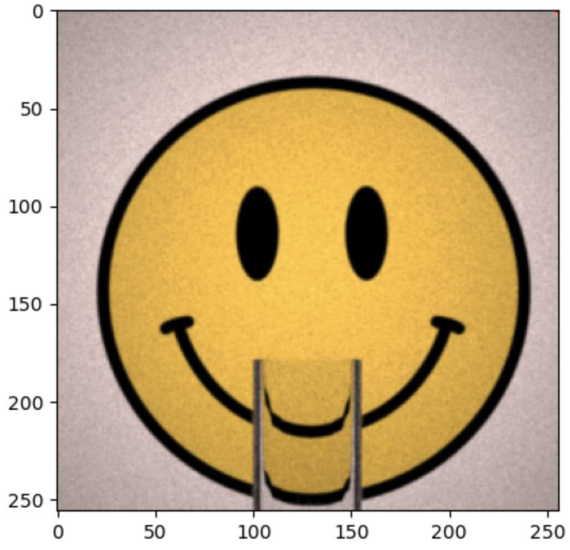
Threshold = 0.25

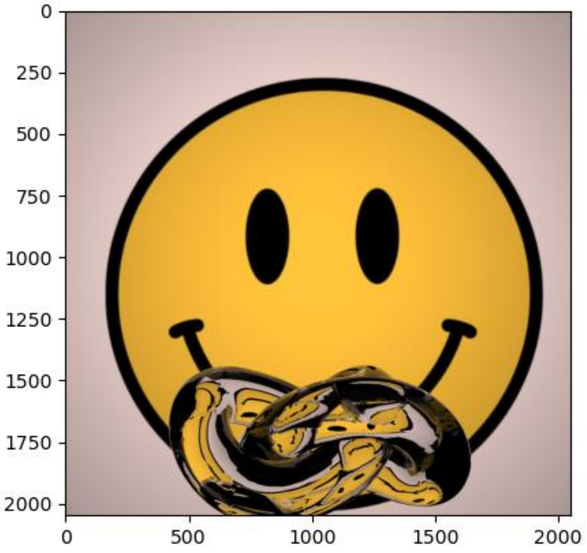
Can we generalize to other shapes?





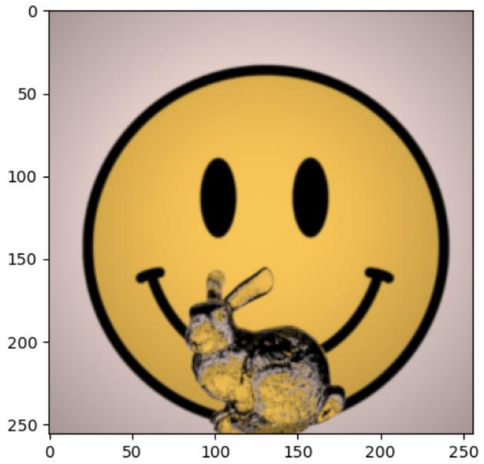


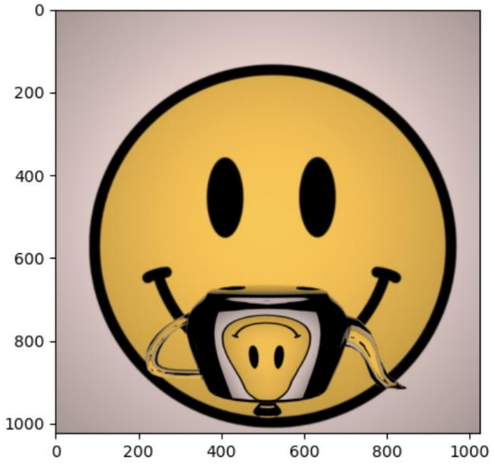


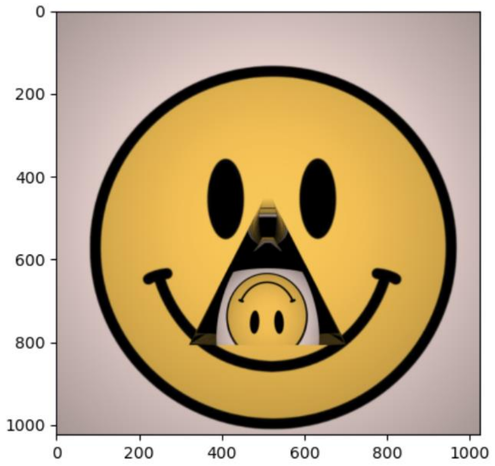


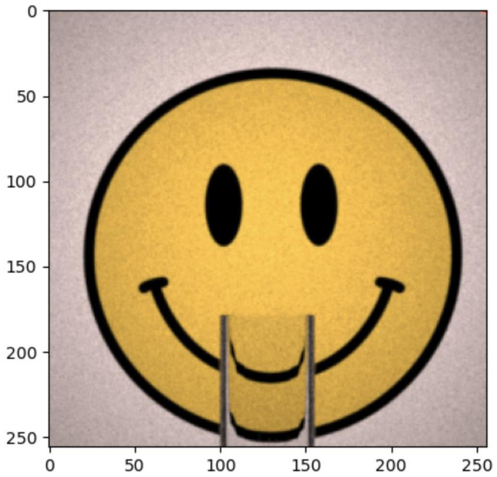
Complicating factors include multiple refractions, scope of the object, and high frequency mapping function

Can we generalize to other shapes?



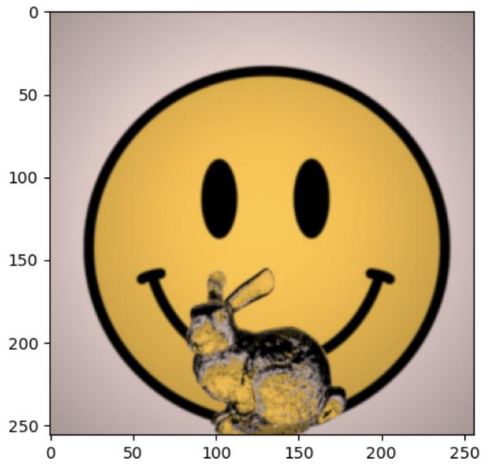


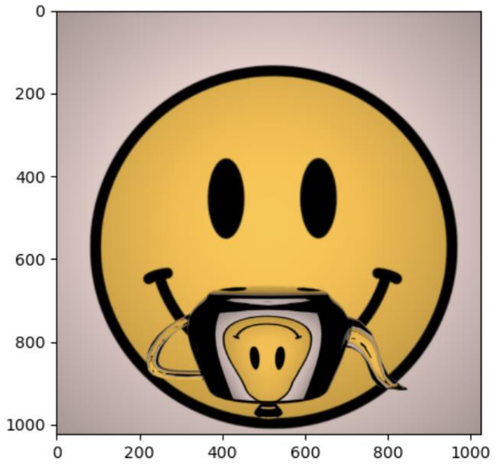


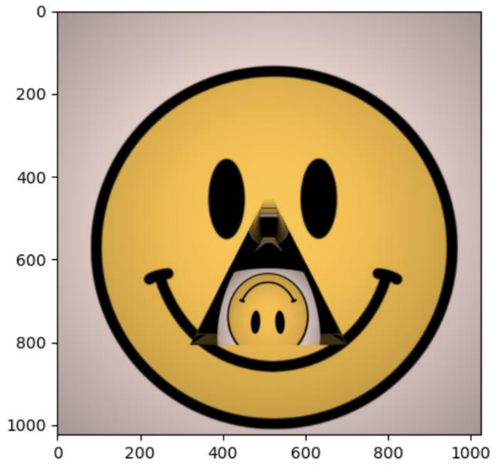


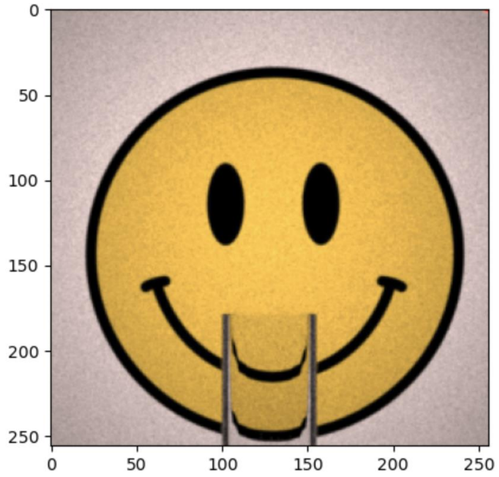
Key considerations:

Can we generalize to other shapes?





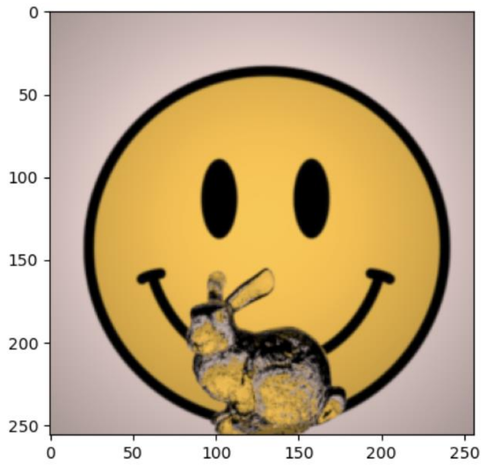


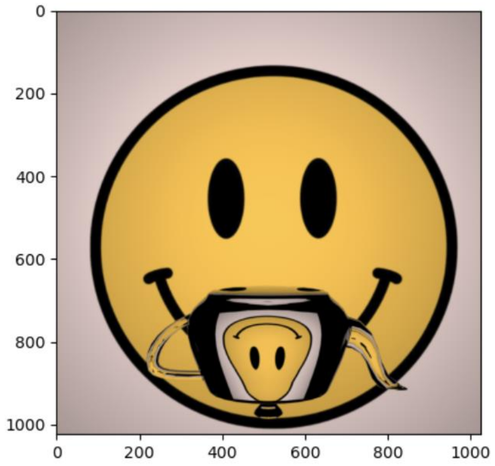


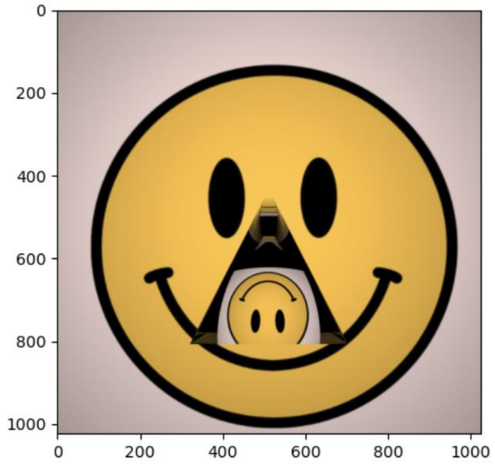
Key considerations:

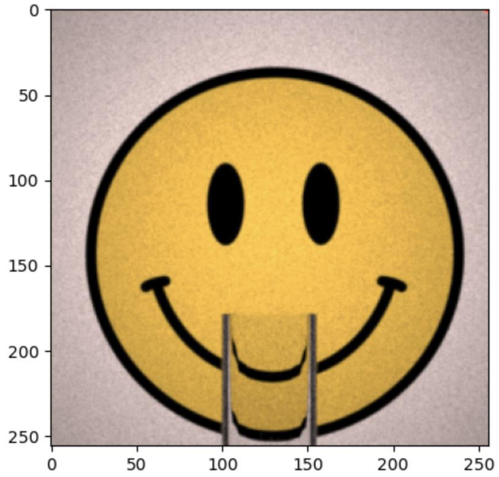
- Does the totem support the full scene?

Can we generalize to other shapes?





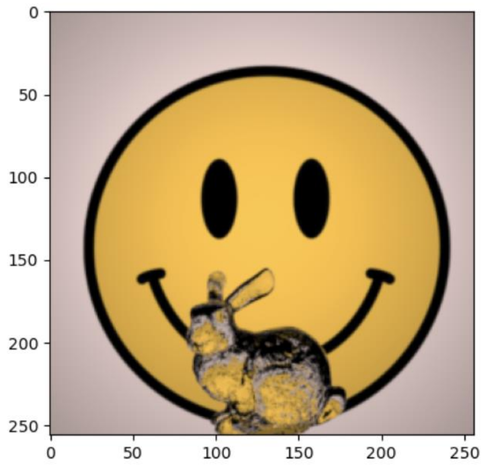


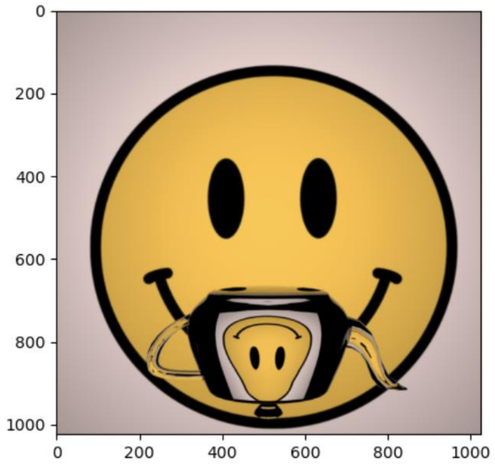


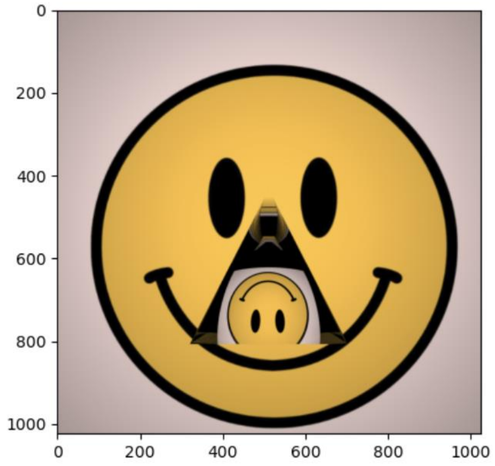
Key considerations:

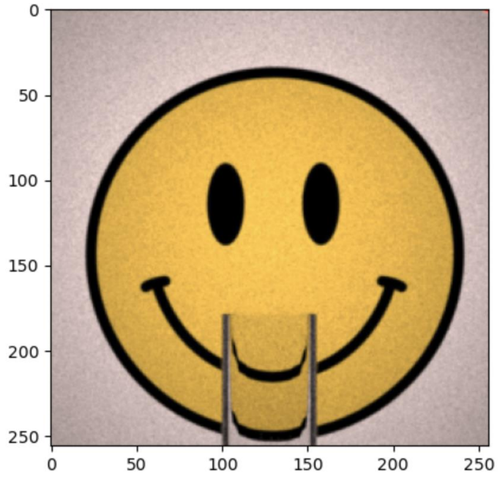
- Does the totem support the full scene?
  - Not necessarily, a sphere has a wide FOV s.t. the full plane of the scene is captured by the back hemisphere
  - Might want to combine multiple totems into an image to cover full scene, or, only protect certain regions

Can we generalize to other shapes?









Key considerations:

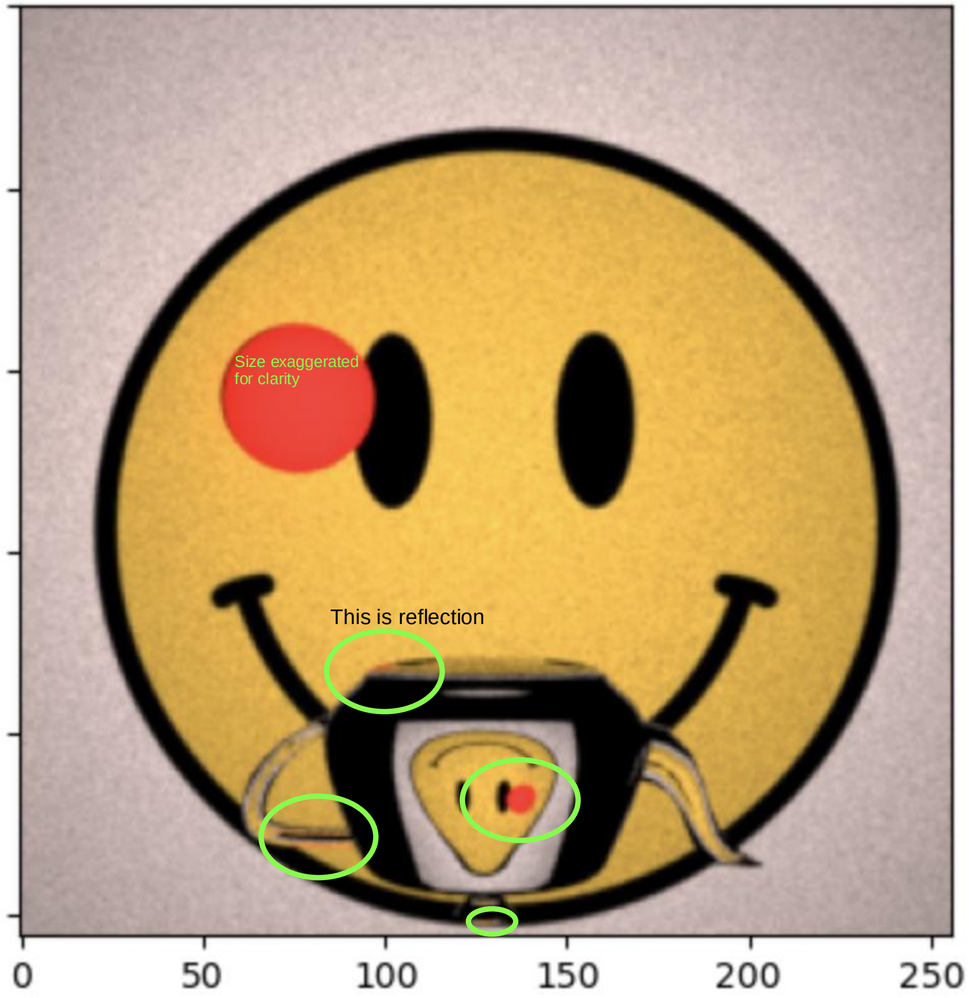
- Does the totem support the full scene?
  - Not necessarily, a sphere has a wide FOV s.t. the full plane of the scene is captured by the back hemisphere
  - Might want to combine multiple totems into an image to cover full scene, or, only protect certain regions
- Is the many:1 $(u_c, v_c):(u_t, v_t)$ mapping ratio preserved

Can we generalize to other shapes?

Key considerations:

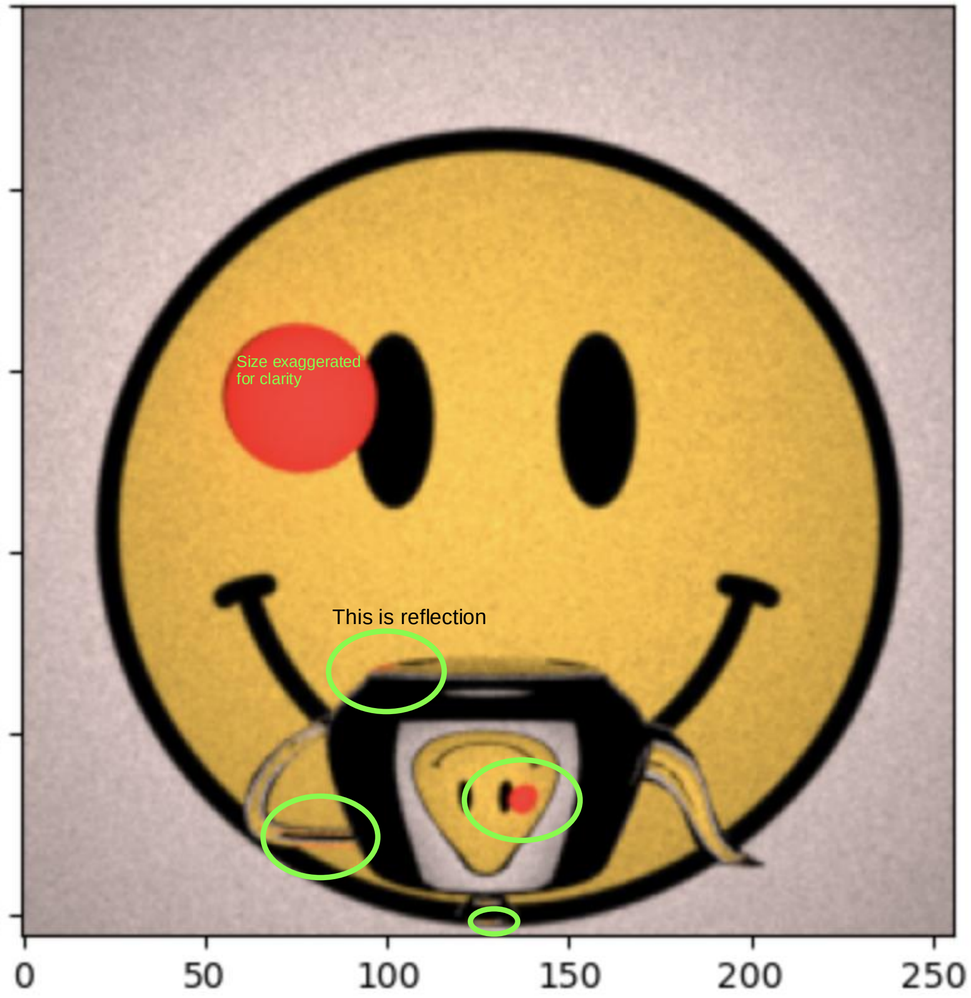
- Does the totem support the full scene?
  - Not necessarily, a sphere has a wide FOV s.t. the full plane of the scene is captured by the back hemisphere
  - Might want to combine multiple totems into an image to cover full scene, or, only protect certain regions
- Is the many:1 $(u_c, v_c):(u_t, v_t)$ mapping ratio preserved
  - Some totems have many:many correspondences due to multiple refractions at a single laser loc

The performance of a “multi-refraction” totem shape

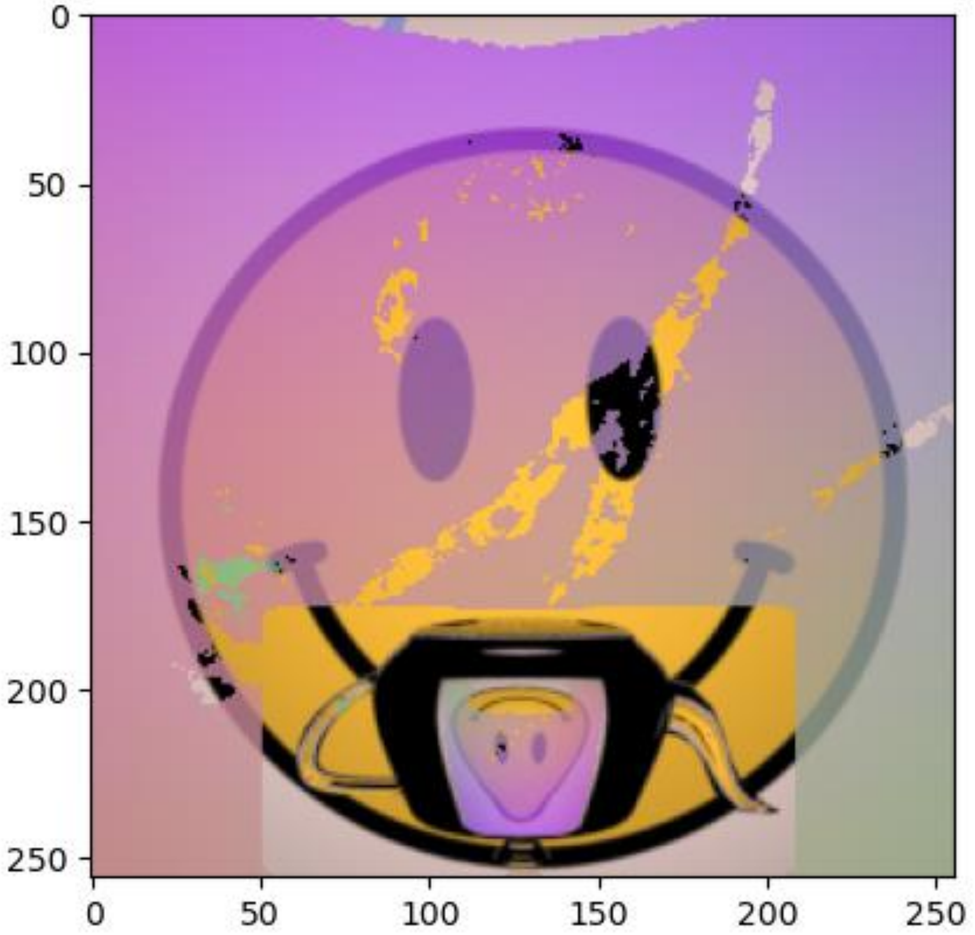


- Notice the presence of red pixels in multiple locations
- Some are reflections, some are refractions, but it is not necessarily known which are which
- How well will our current mappings method (discarding up to 2 dimmer clusters of red pixels) perform here?

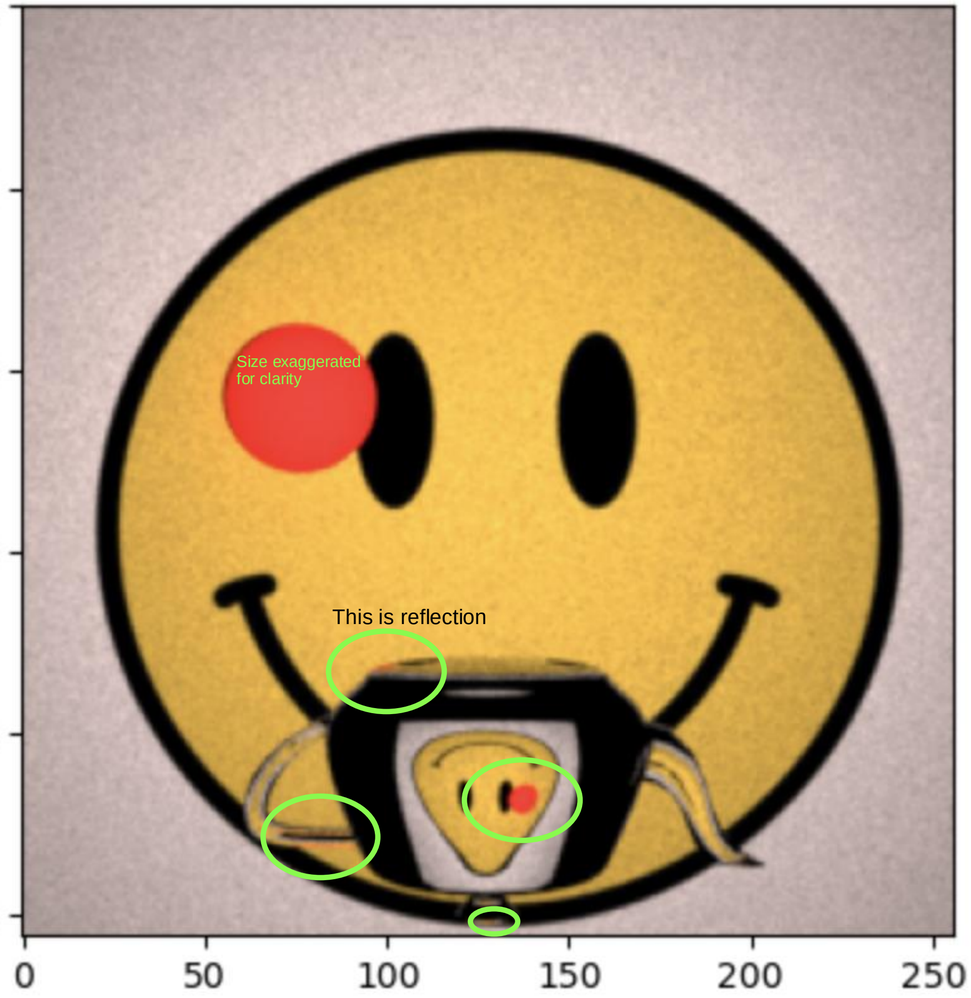
The performance of a “multi-refraction” totem shape



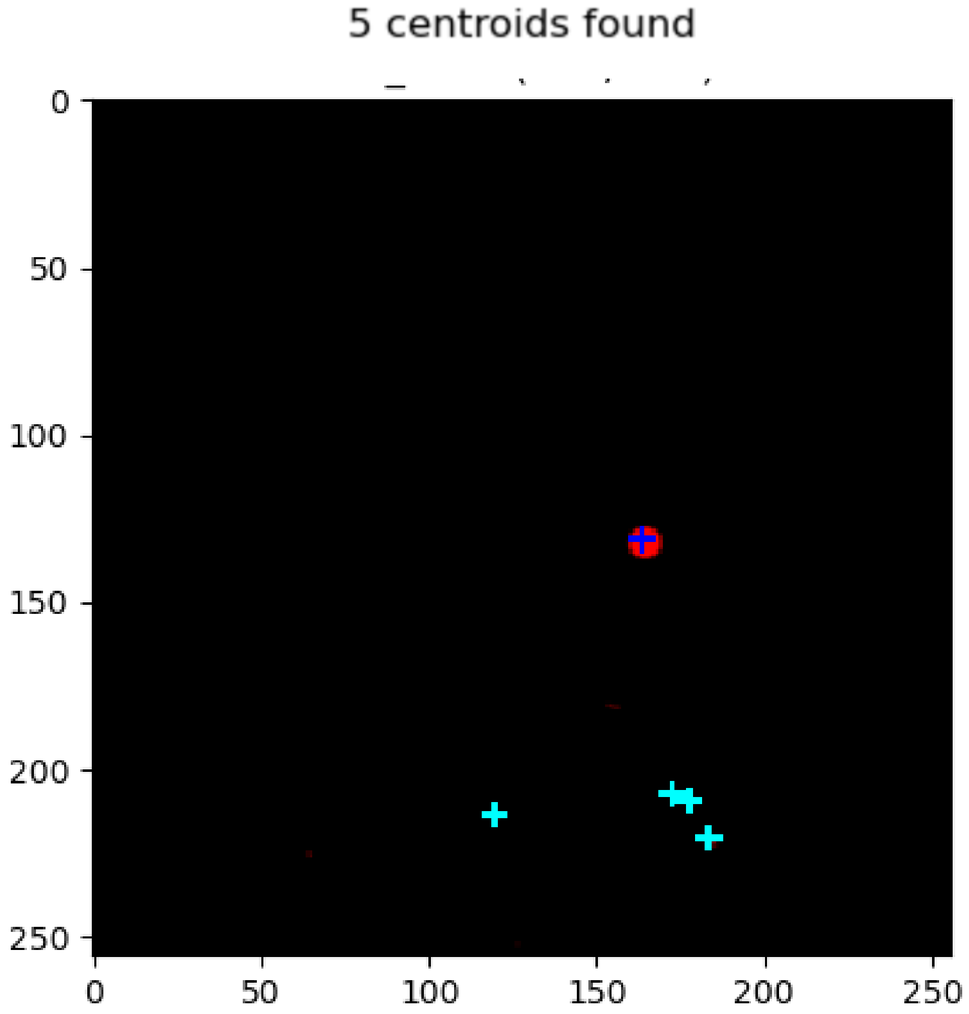
- Notice the presence of red pixels in multiple locations
- Some are reflections, some are refractions, but it is not necessarily known which are which
- How well will our current mappings method (discarding up to 2 dimmer clusters of red pixels) perform here?



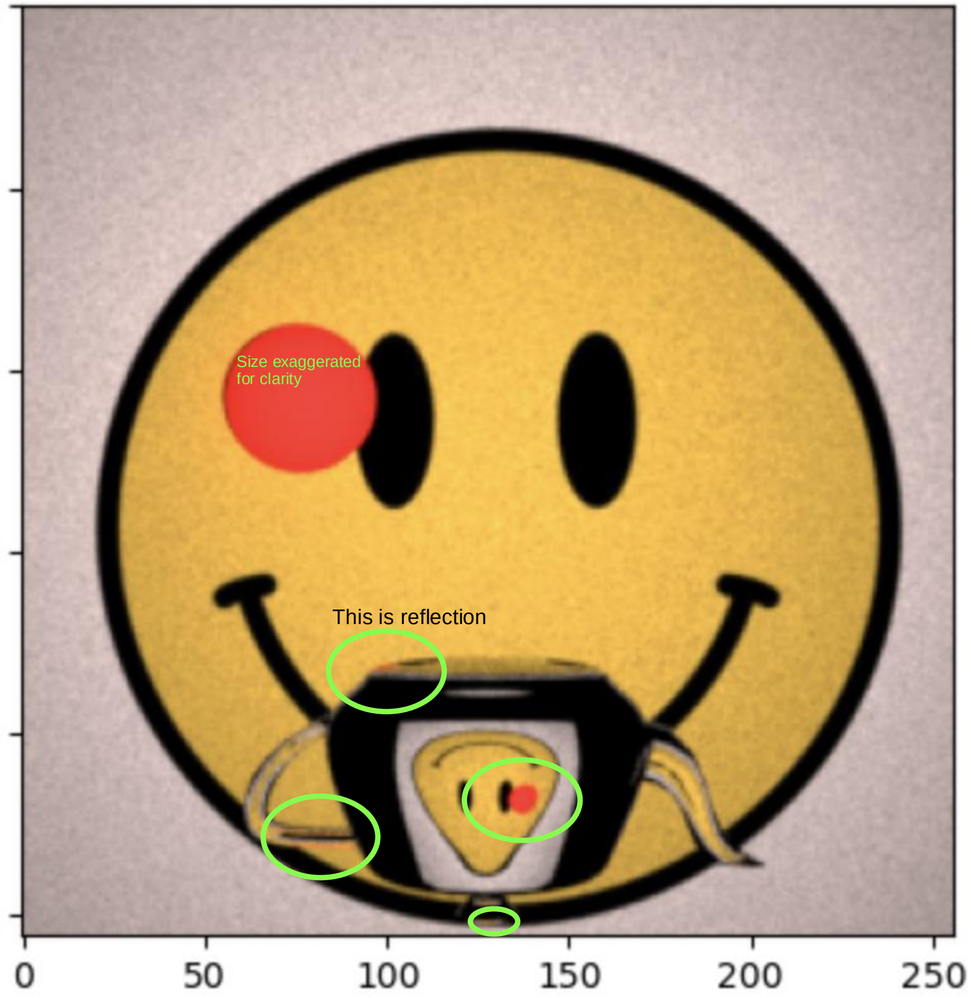
The performance of a “multi-refraction” totem shape



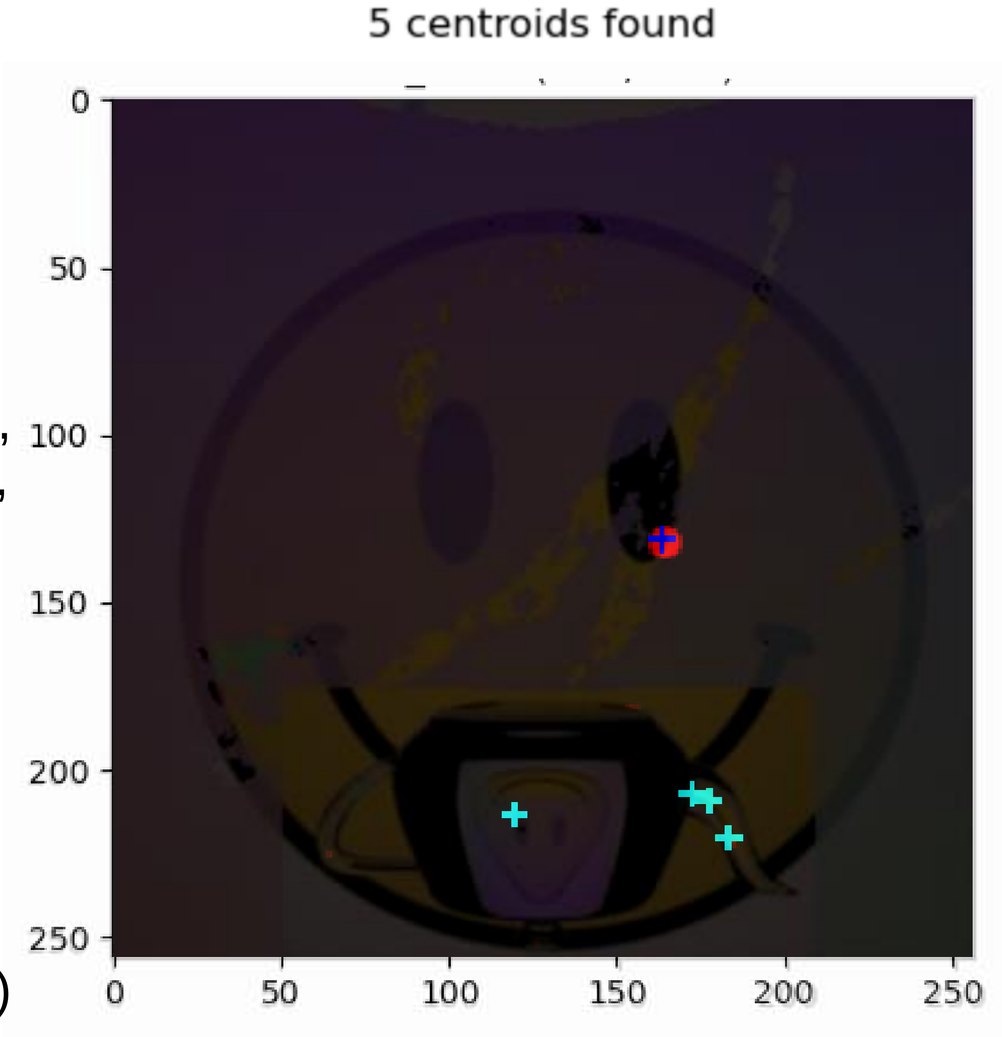
- Notice the presence of red pixels in multiple locations
- Some are reflections, some are refractions, but it is not necessarily known which are which
- How well will our current mappings method (discarding up to 2 dimmer clusters of red pixels) perform here?



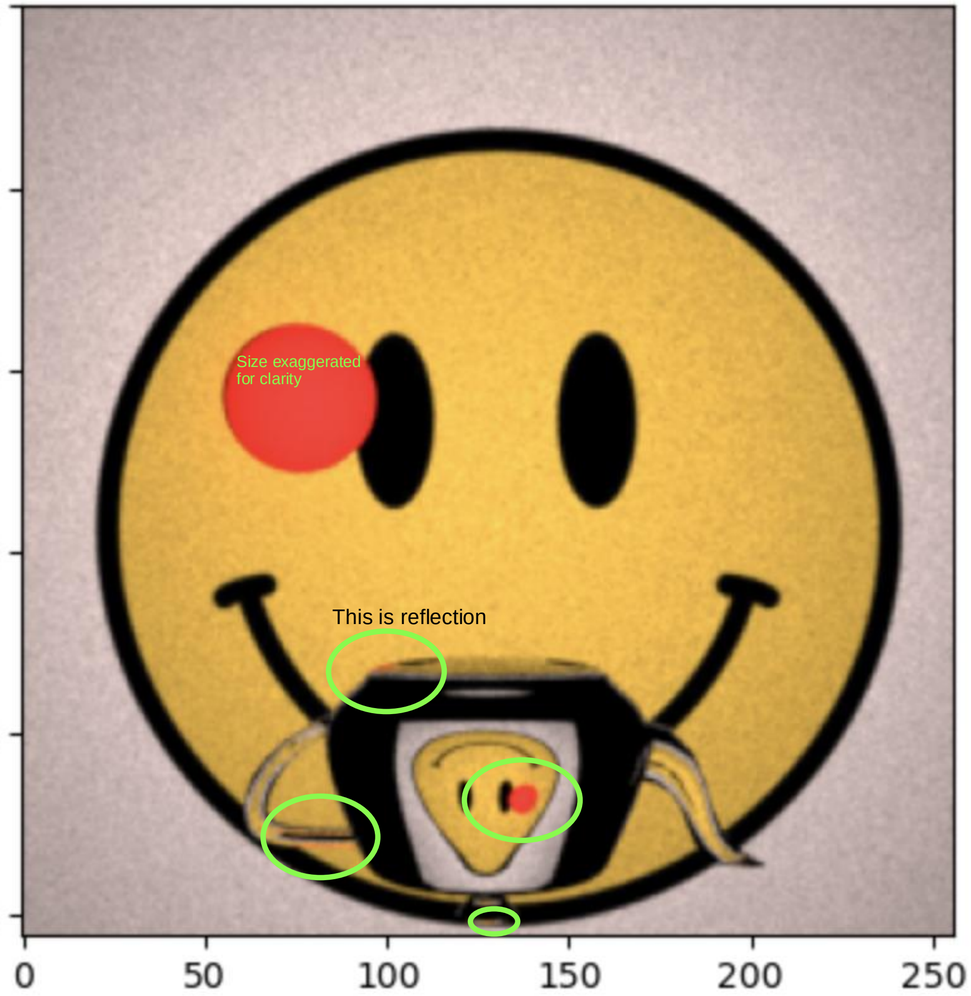
The performance of a “multi-refraction” totem shape



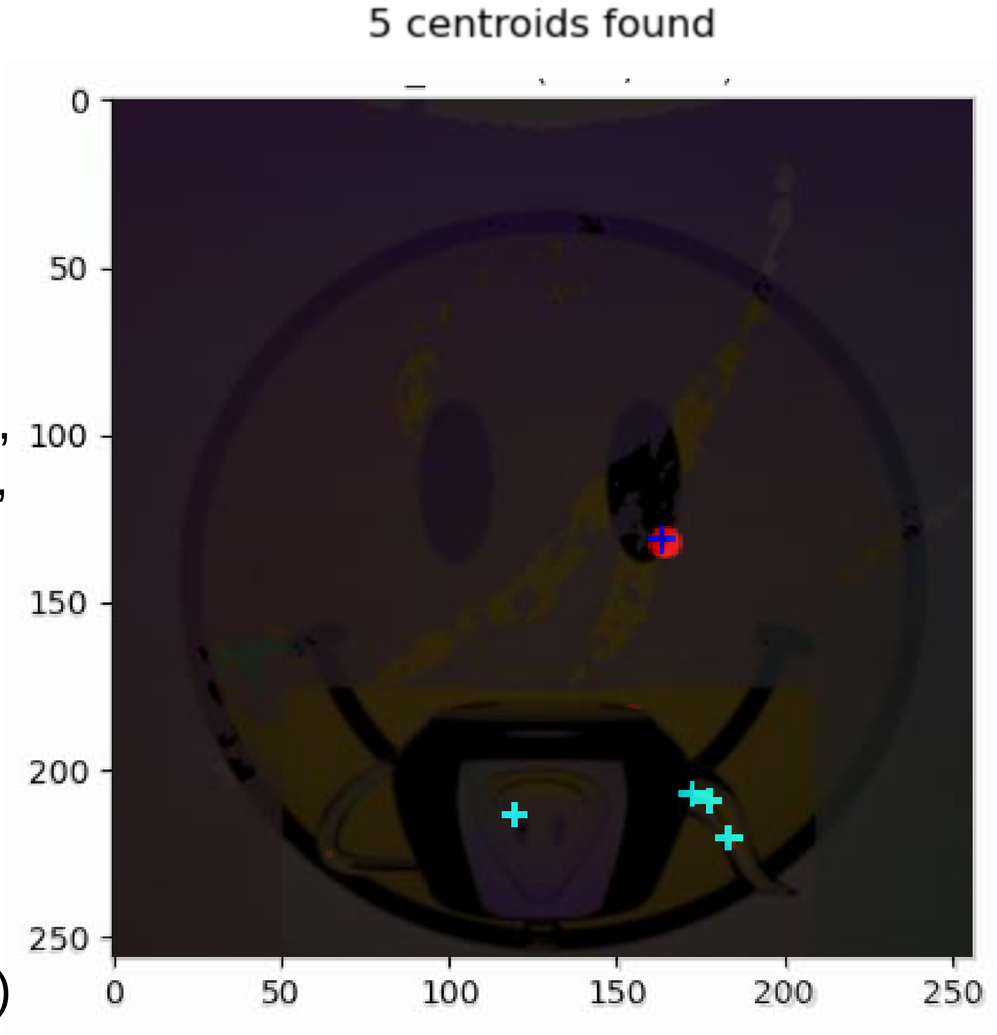
- Notice the presence of red pixels in multiple locations
- Some are reflections, some are refractions, but it is not necessarily known which are which
- How well will our current mappings method (discarding up to 2 dimmer clusters of red pixels) perform here?



The performance of a “multi-refraction” totem shape



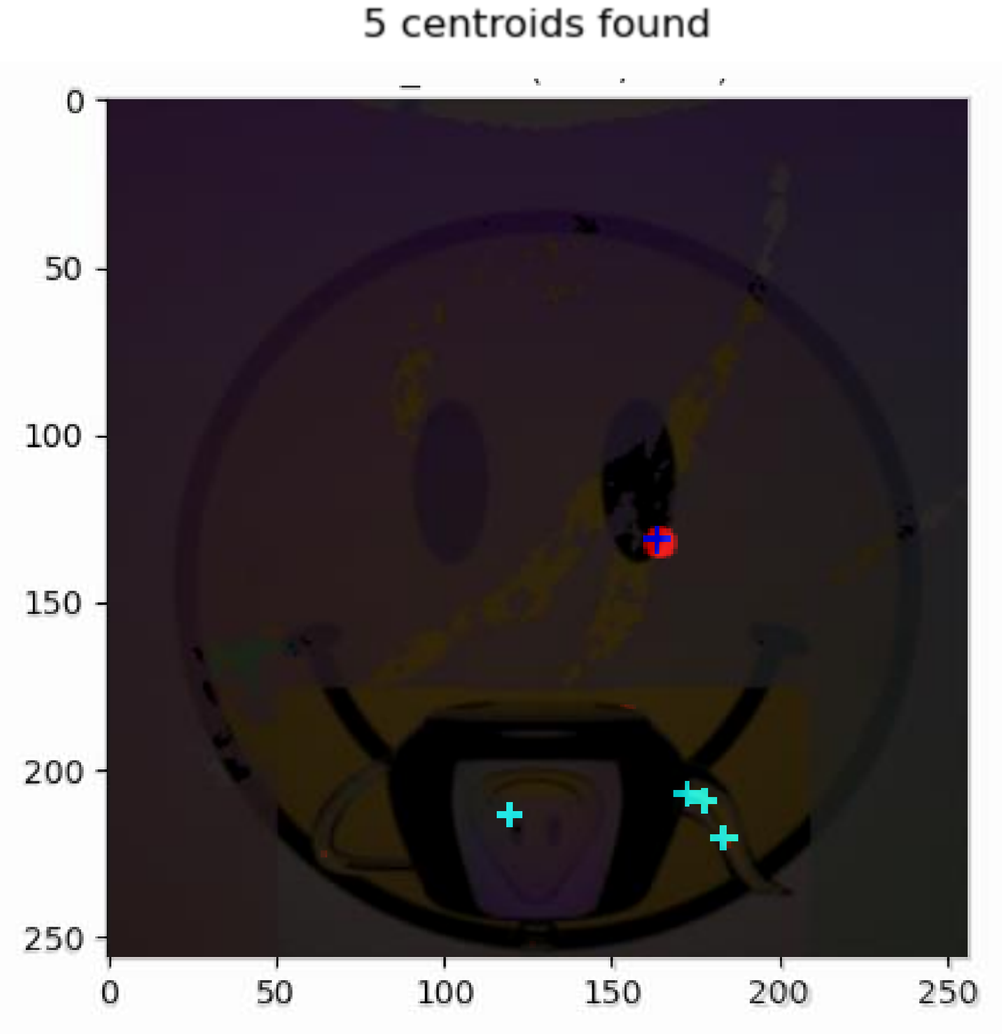
- Notice the presence of red pixels in multiple locations
- Some are reflections, some are refractions, but it is not necessarily known which are which
- How well will our current mappings method (discarding up to 2 dimmer clusters of red pixels) perform here?



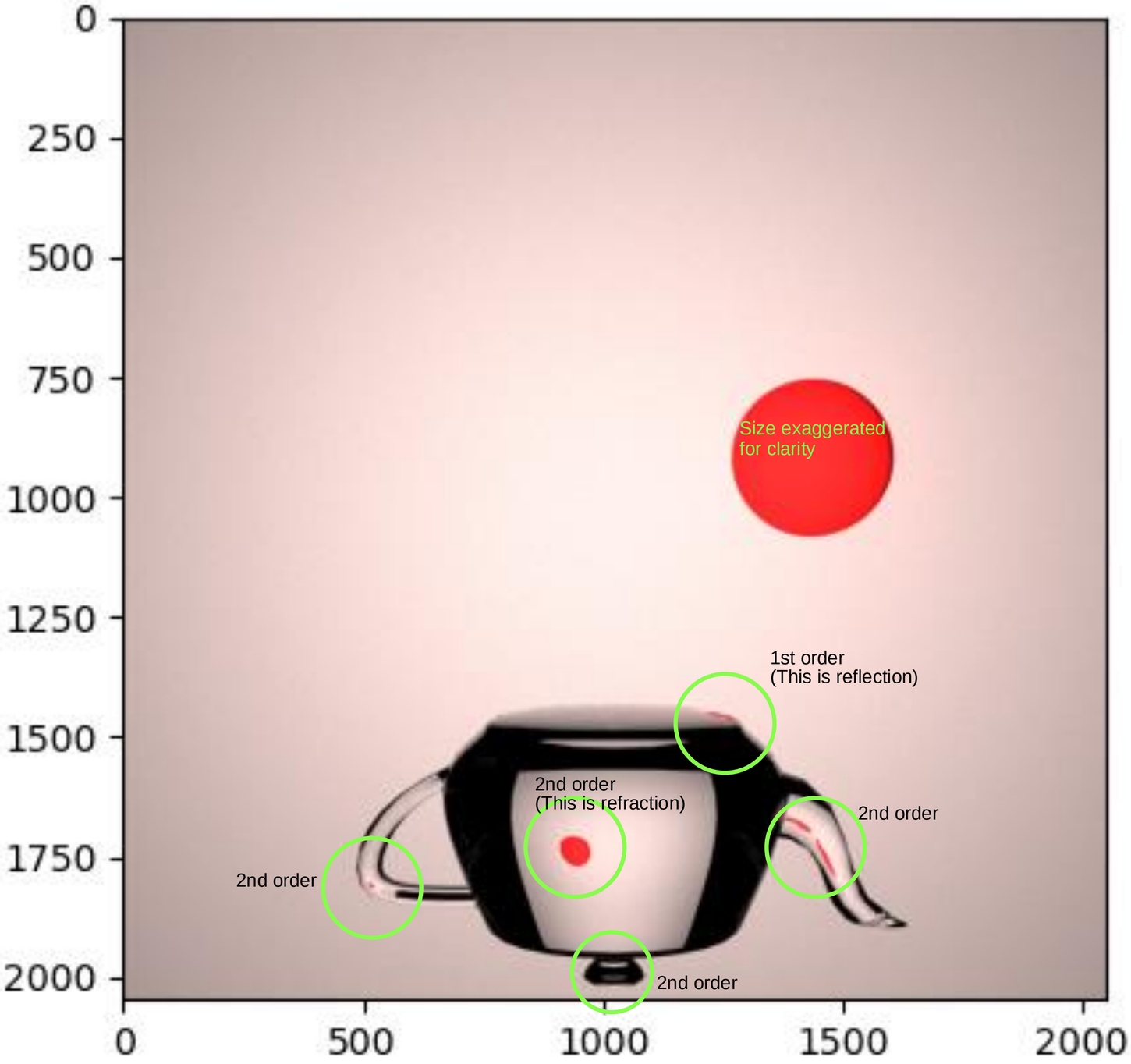
How well will the MLP interpolate for the failed points?

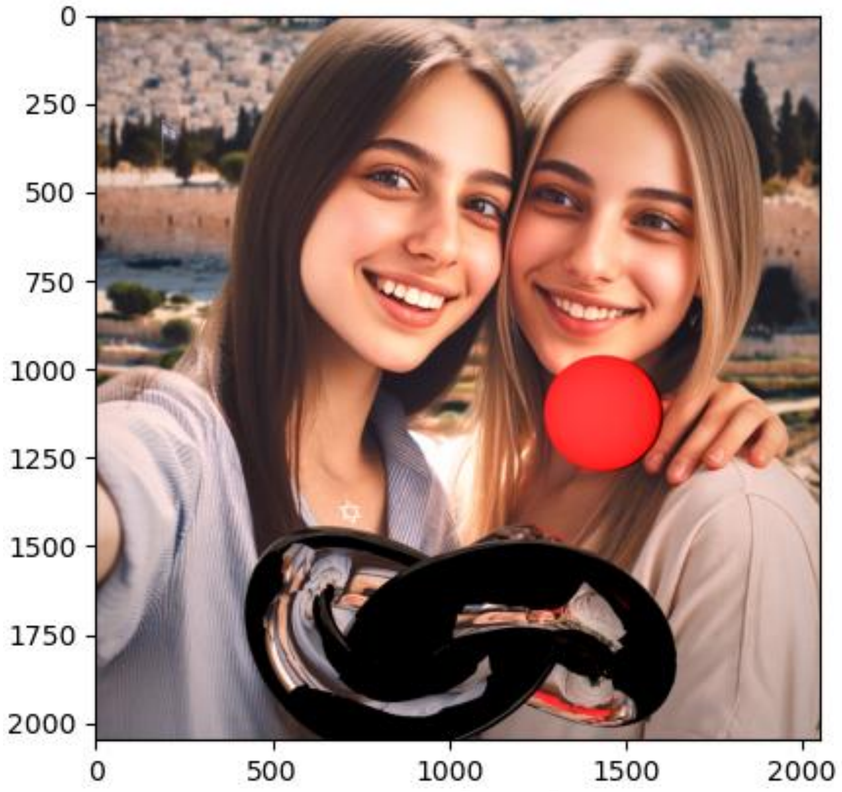
The performance of a “multi-refraction” totem shape

- Option 1: Use the current many:1 data and an MLP similar to the sphere architecture
- Option 2: Change the mappings procedure to save all refractions found and use a VAE to represent the distribution of mappings

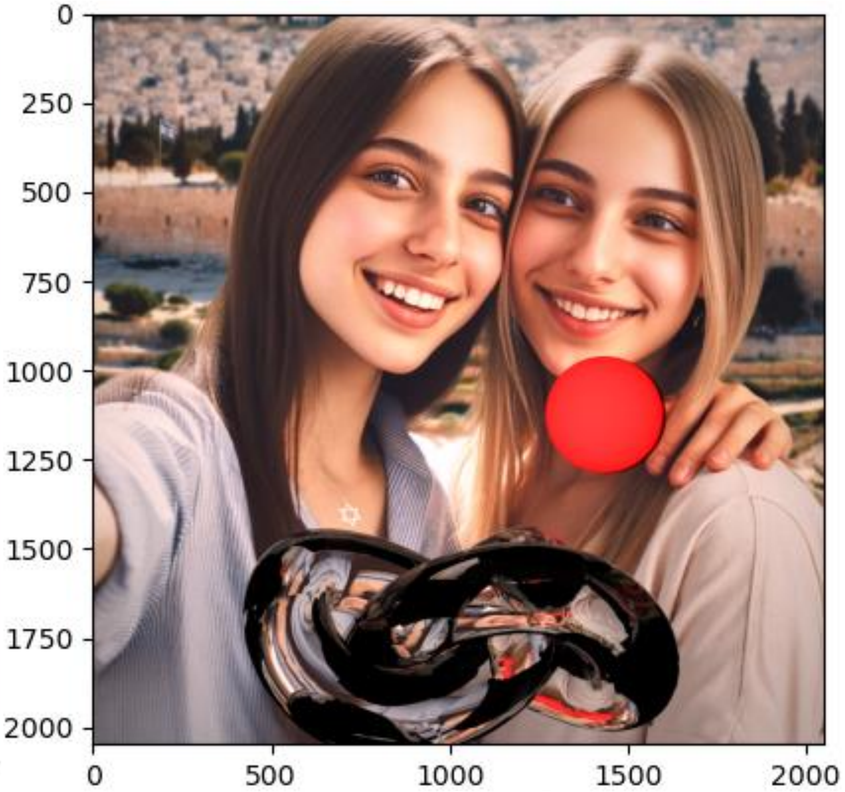


How well will the MLP interpolate for the failed points?

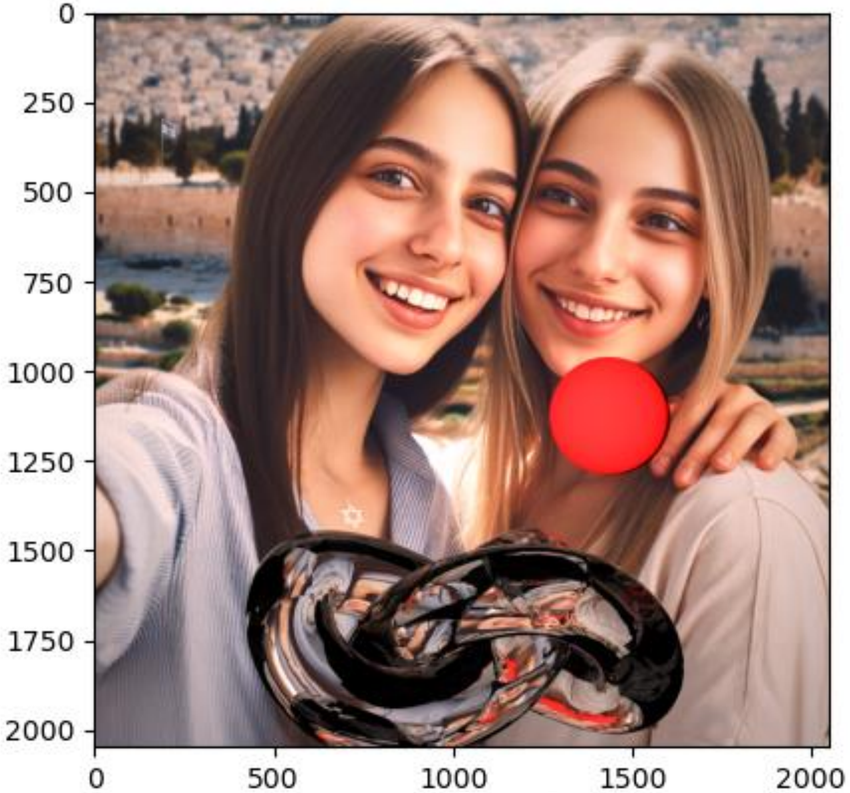

0
250
500
750
1000
1250
1500
1750
2000
0
500
1000
1500
2000
Size exaggerated
for clarity
1st order
(This is reflection)
2nd order
(This is refraction)
2nd order
2nd order
2nd order



max_depth 4



max_depth 6



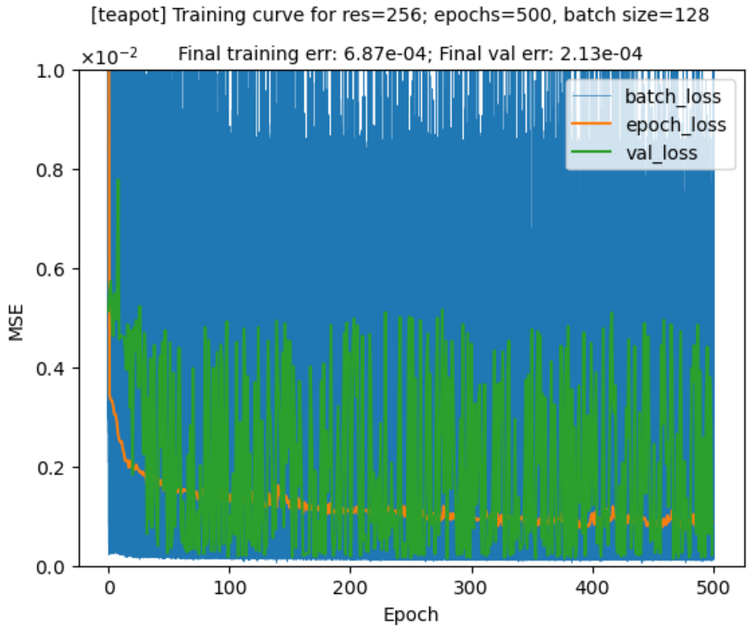
max_depth 9

Positional encodings for multi-refrac shapes

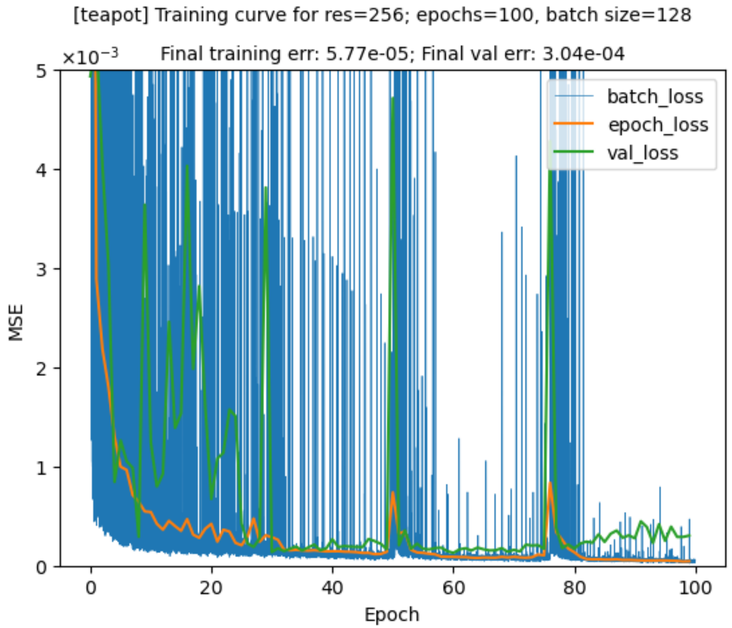
Ablation study

- Due to the possible sharp fluctuations in the mappings function for a multi-refraction shape like a teapot, best to use positional encodings to represent the data

without positional encoding



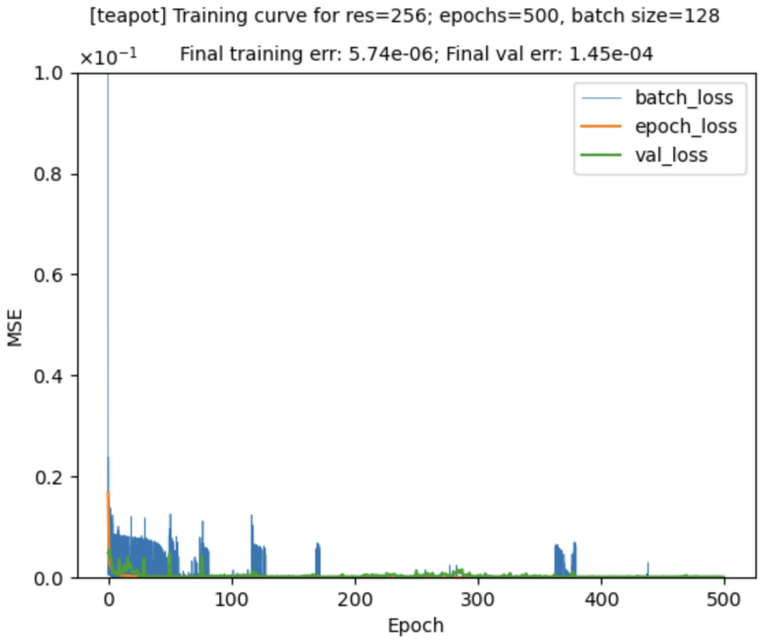
with positional encoding



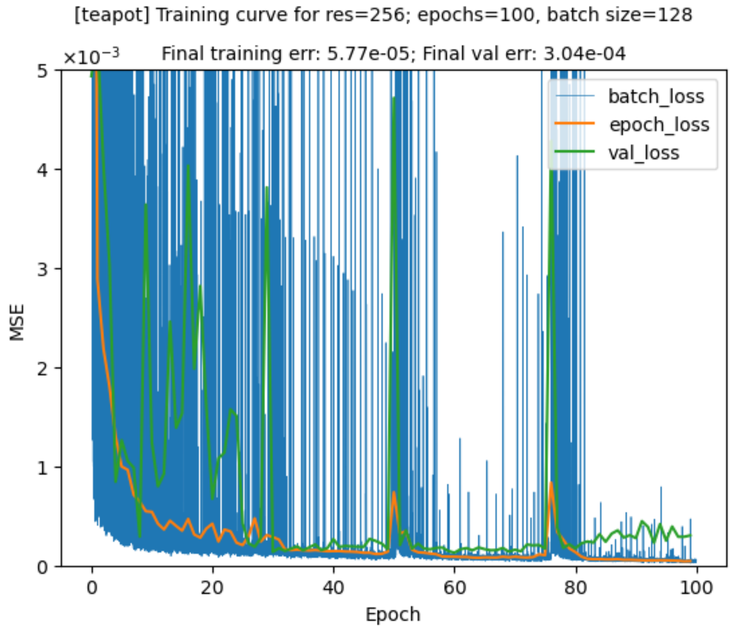
Takes much longer to optimize; can’t achieve as low a test error (9.77e-04 vs 1.23e-03).
Lots of noise in the optimization

Scaled training curve teapot

without positional encoding



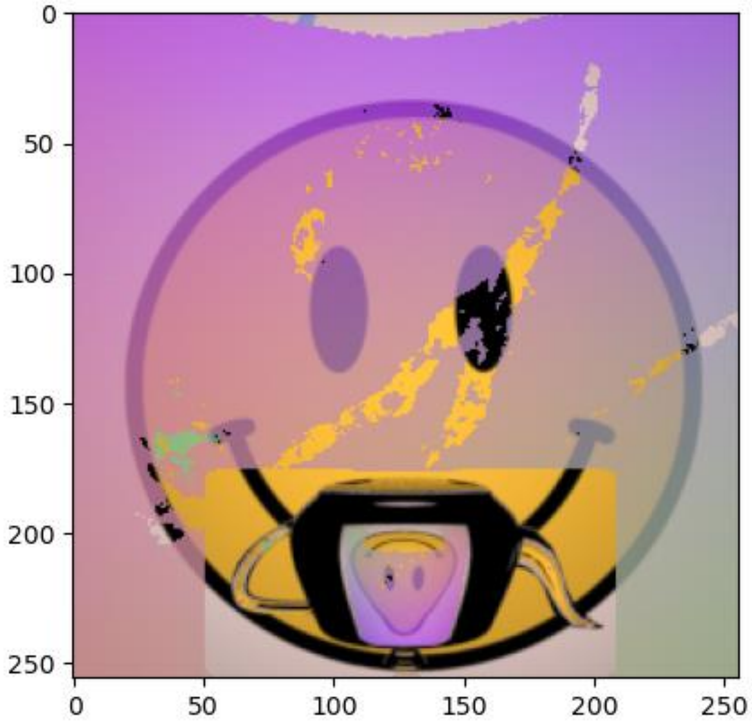
with positional encoding

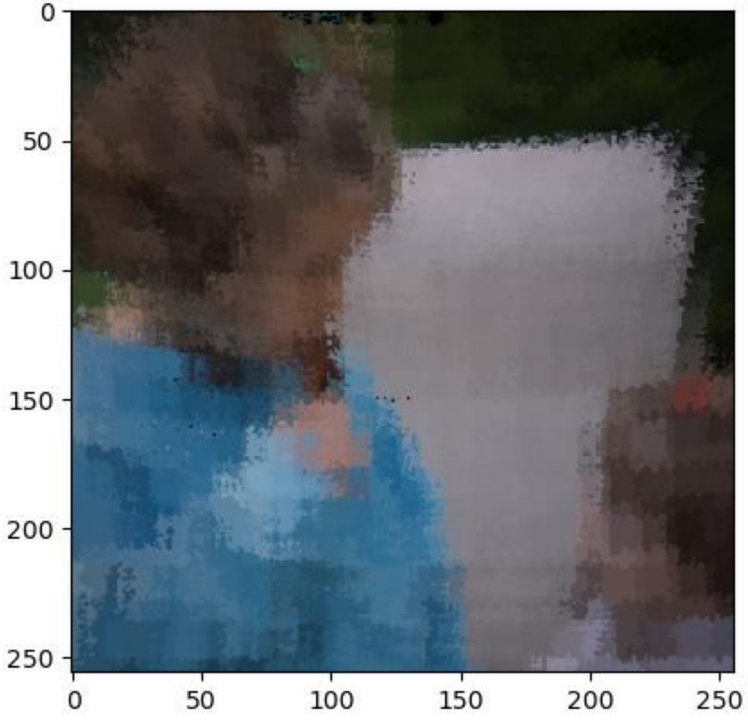


Positional encodings for multi-refrac shapes

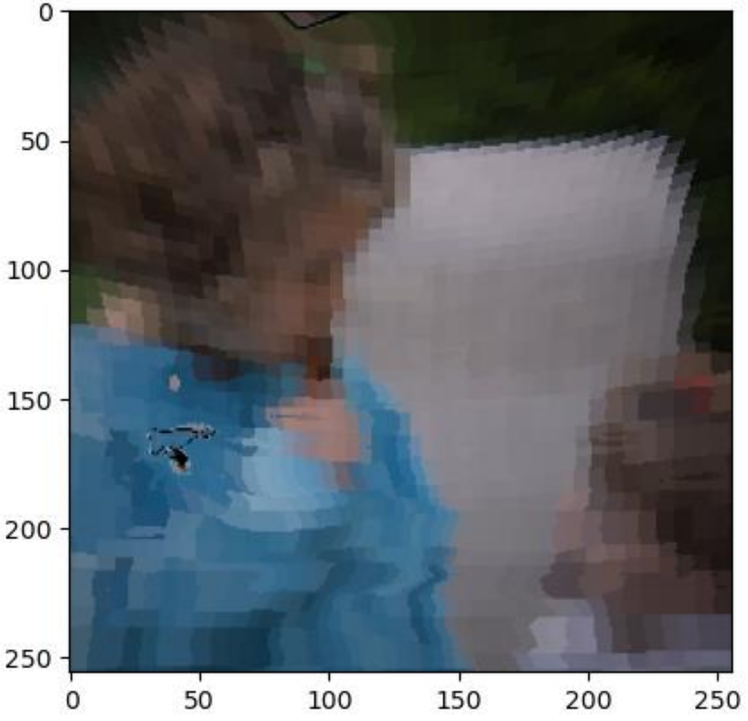
Ablation study

- Due to the possible sharp fluctuations in the mappings function for a multi-refraction shape like a teapot, best to use positional encodings to represent the data





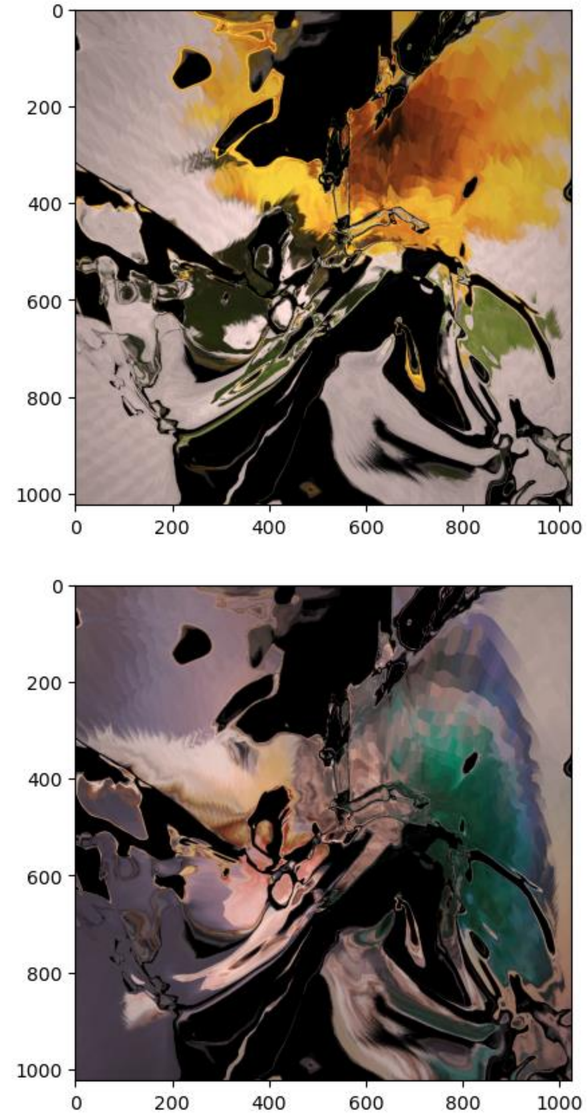
w/ embeddings



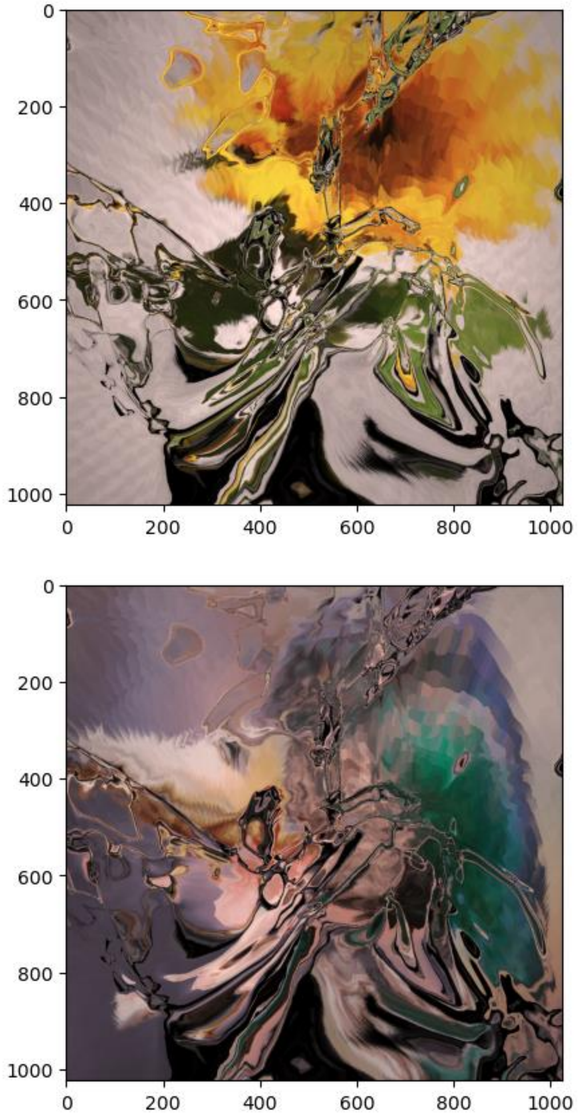
w/o embeddings: artifact of high frequency but smoother?

Ablation studies - positional encodings + max depth

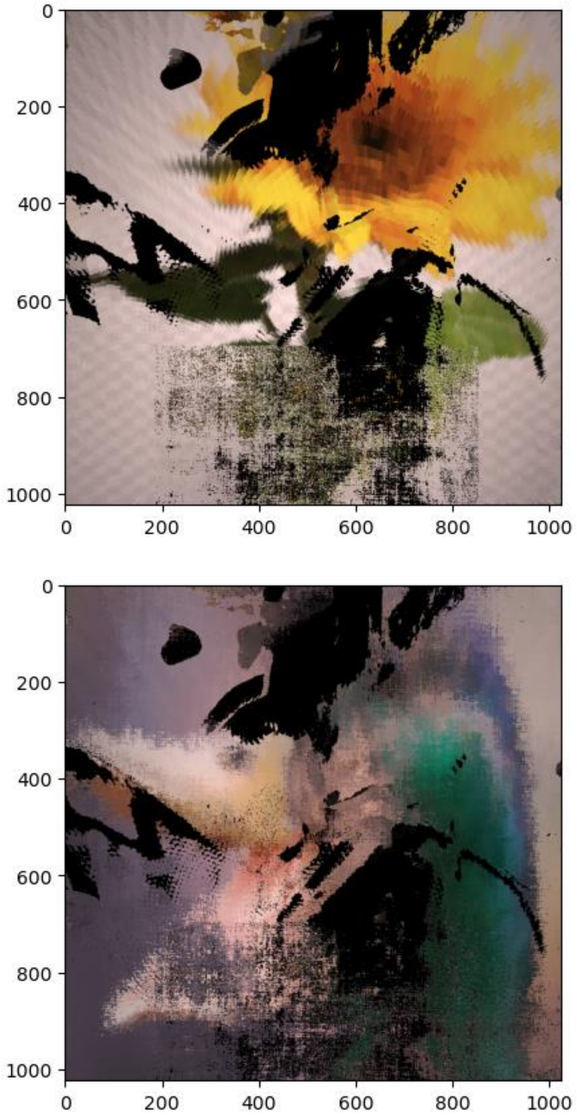
Without positional encodings
Without increased max depth



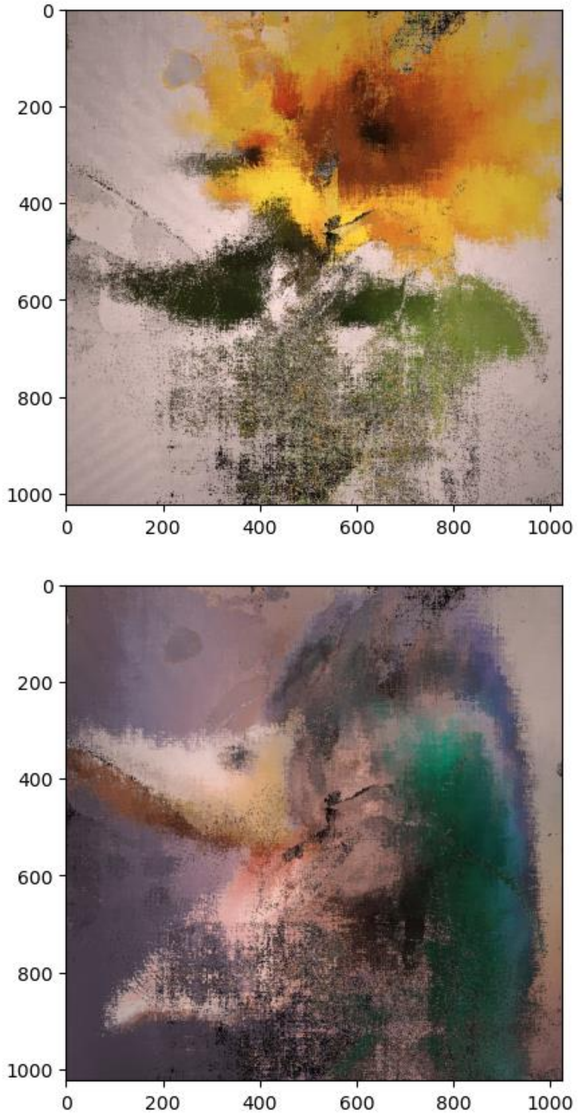
Without positional encodings
Increased max depth



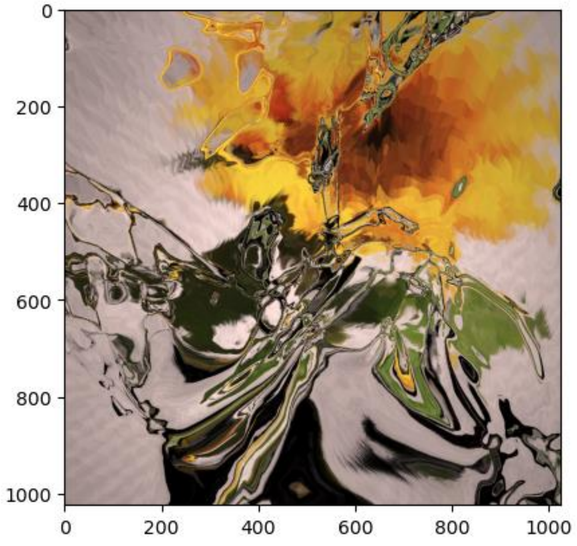
With positional encodings
Without increased max depth



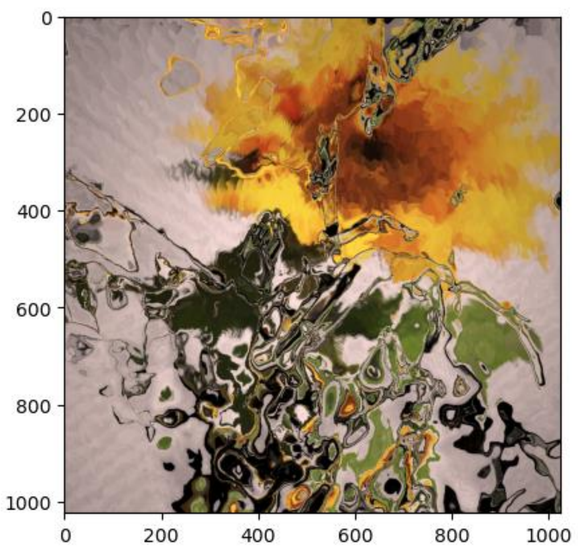
With positional encodings
Increased max depth



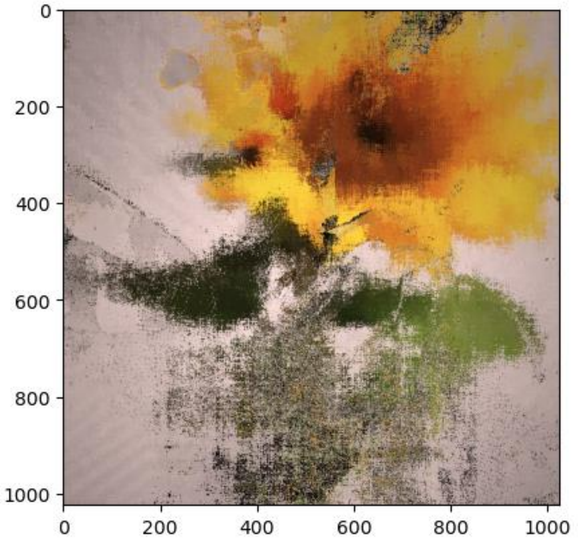
No positional encodings



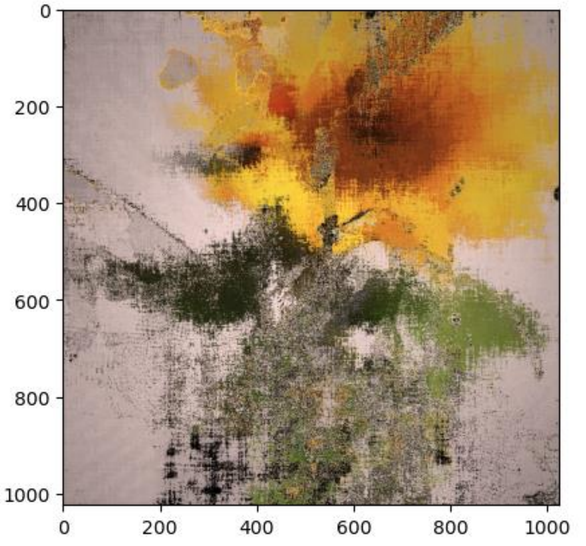
5 frequencies



10 frequencies



20 frequencies

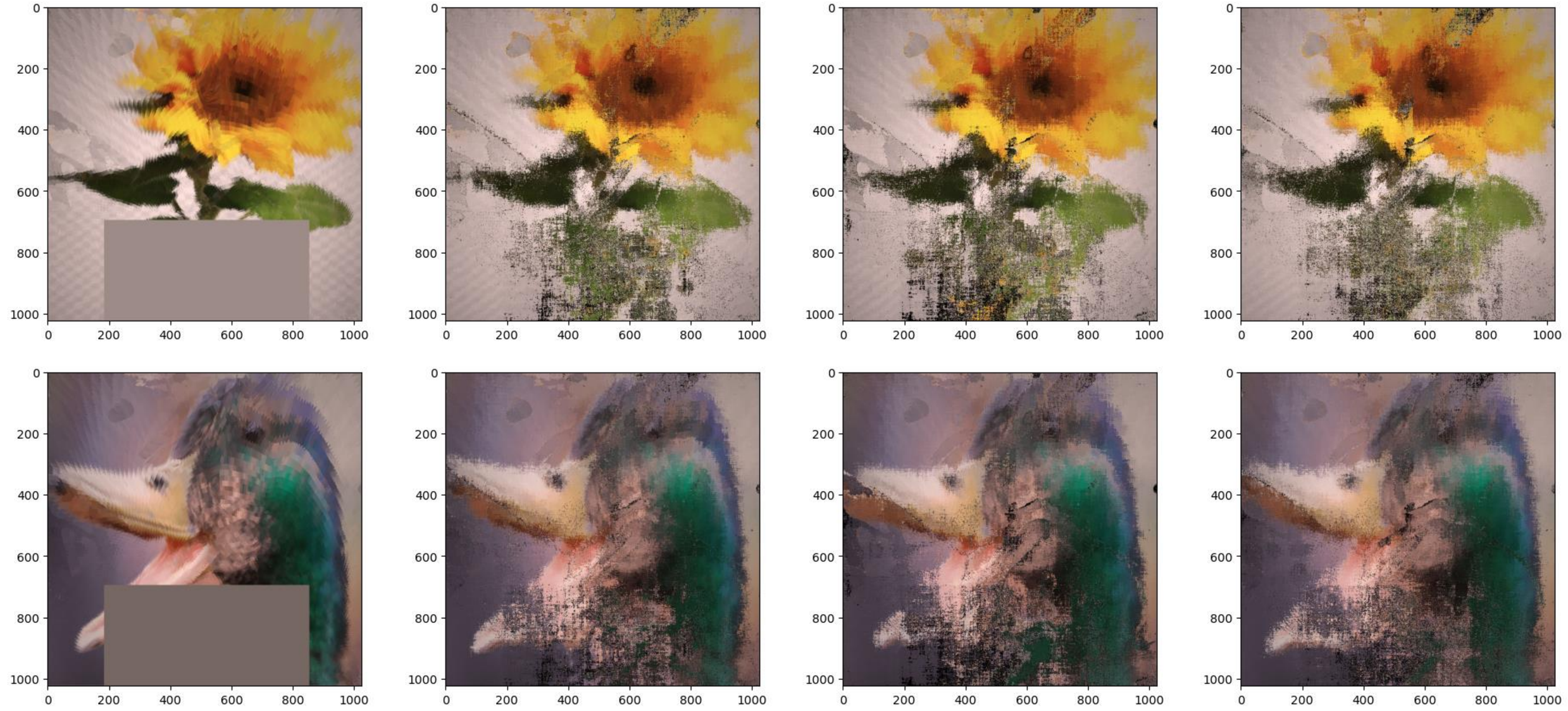


Original Dataset

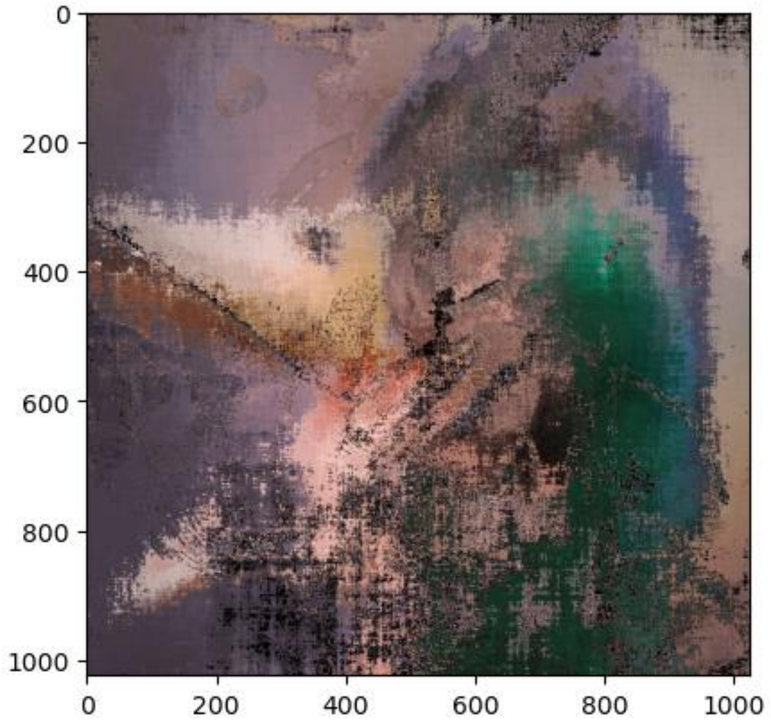
Trained at g=0.30

Trained at g=0.50

Trained at g=0.80

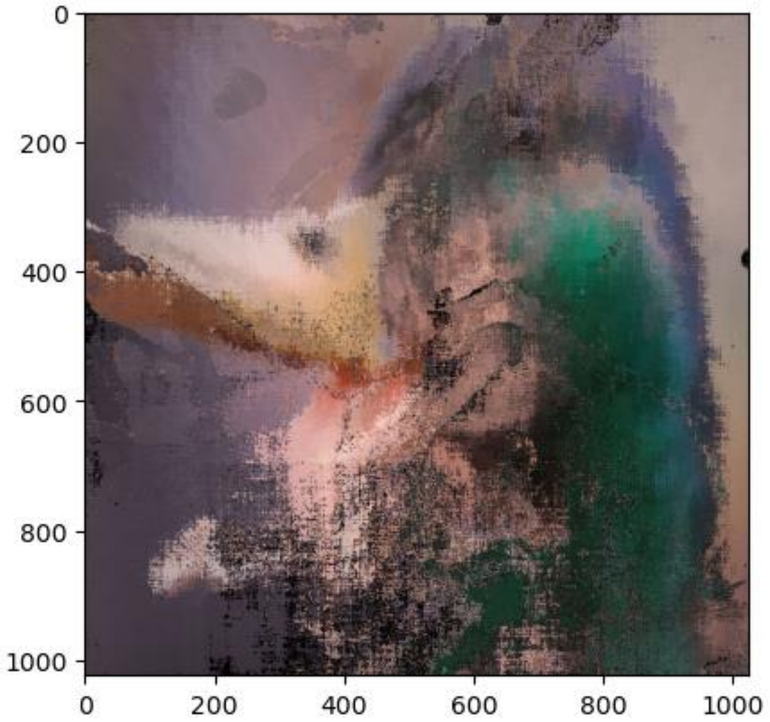


(NN width doubled)



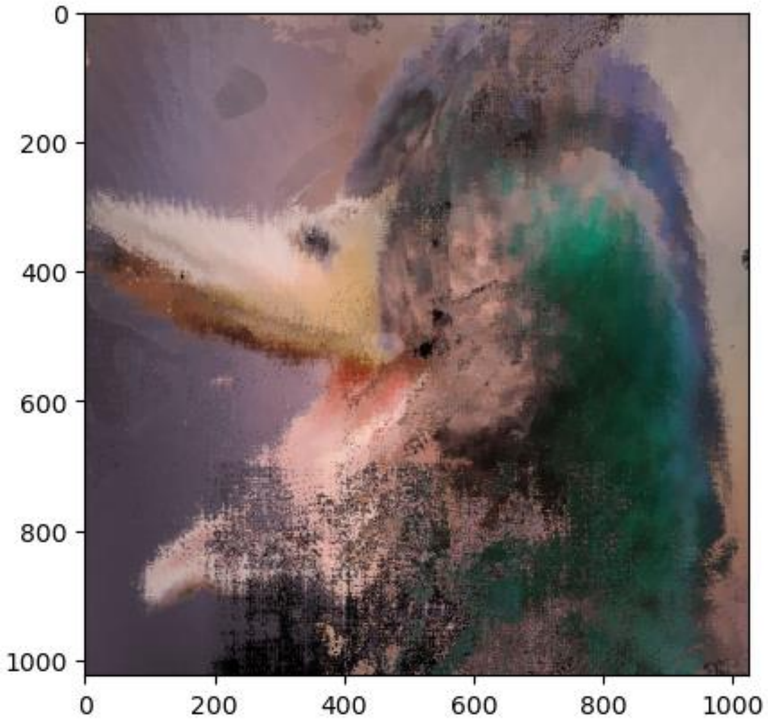
NN width = original (256)

PSNR: 15.14



NN width = 512

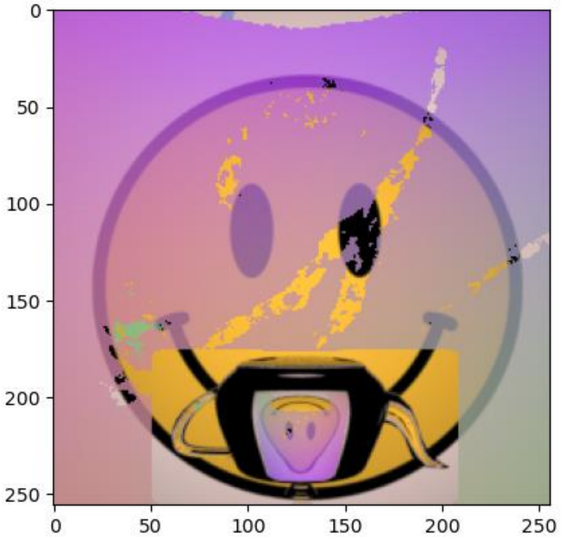
16.02

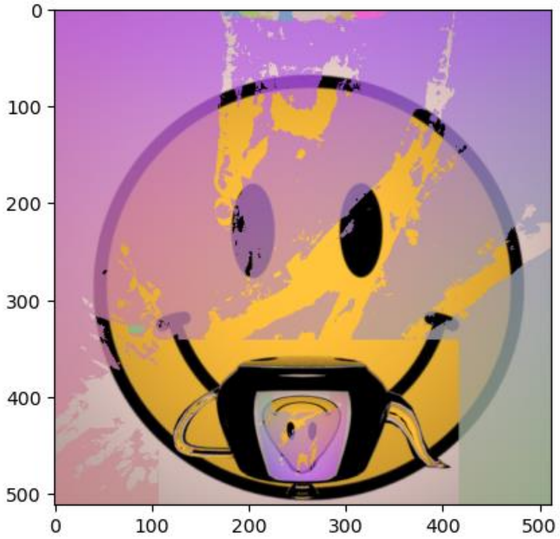


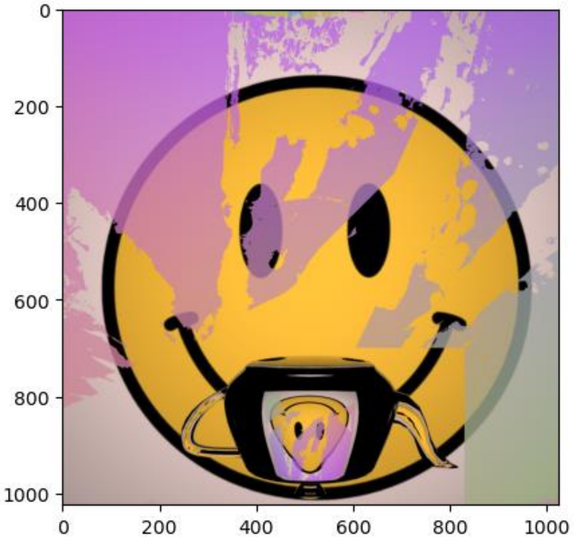
NN width = 1024

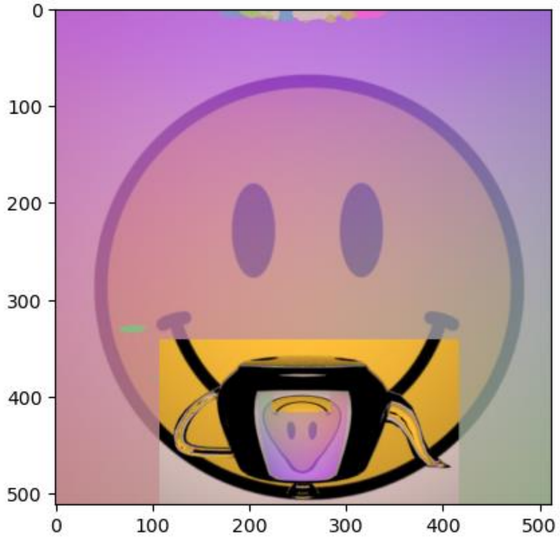
16.56

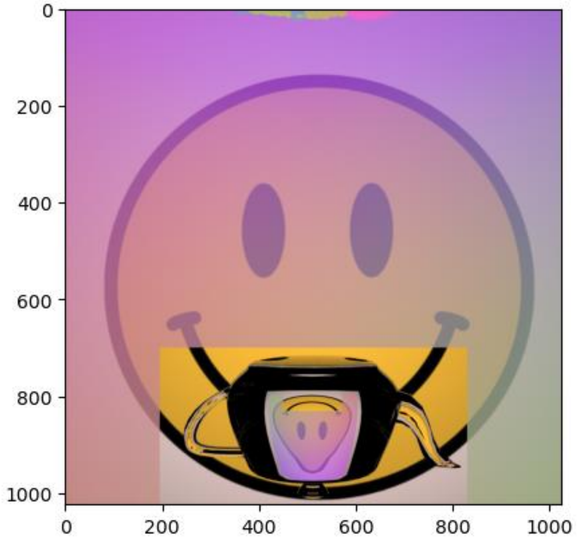
Different dataset variants depending on sampling algorithm & heuristics





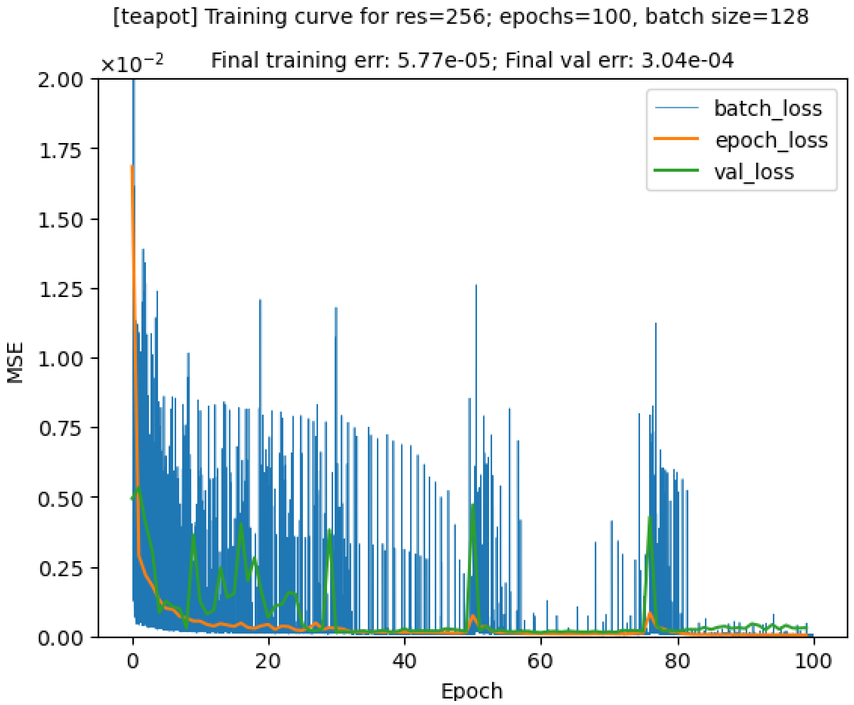


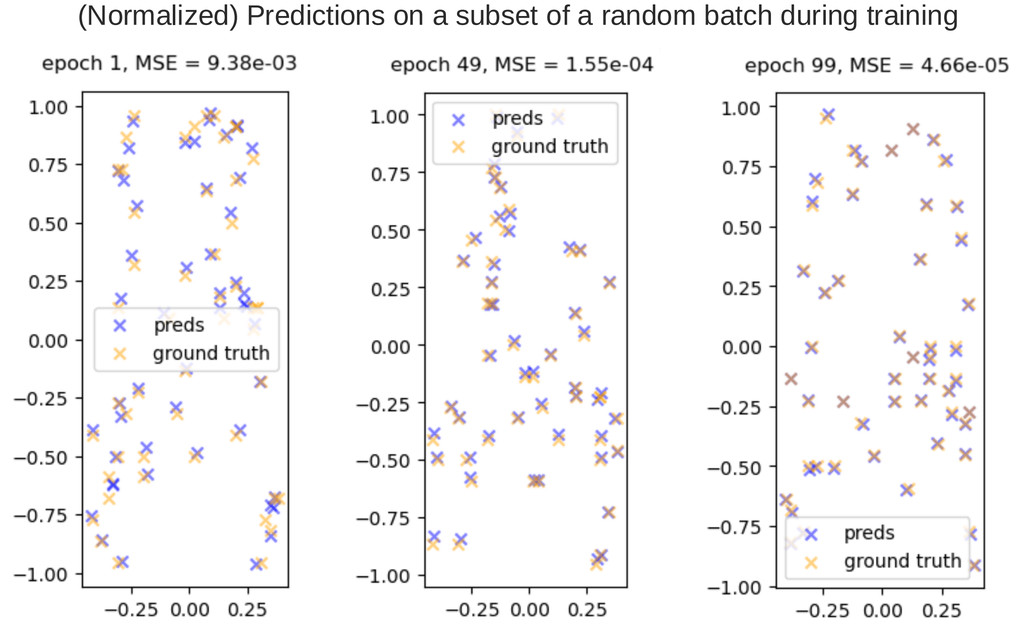


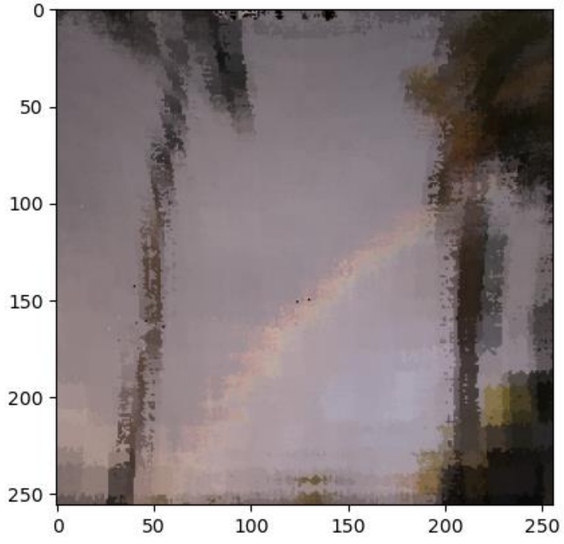


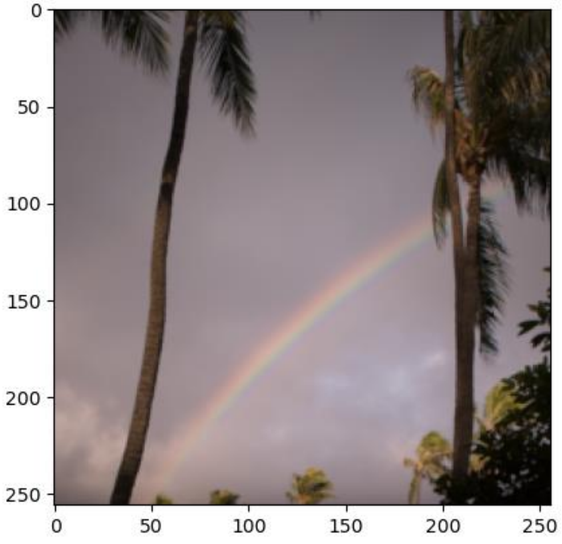
Training + reconstruction results (r=256)

Teapot

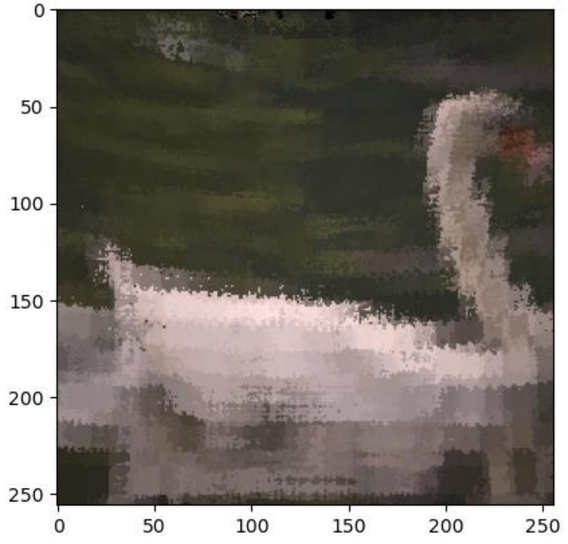


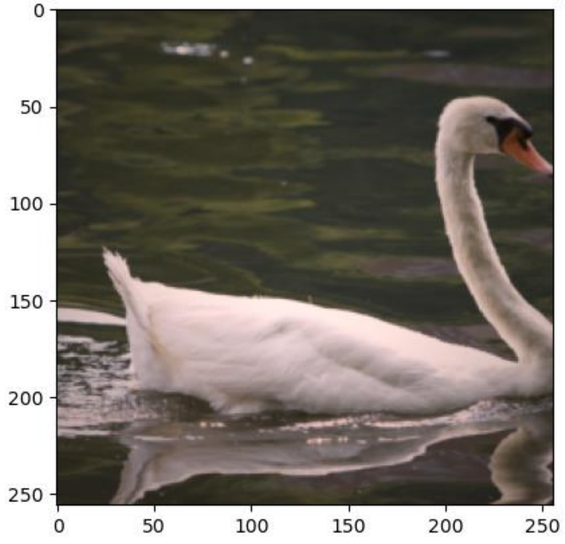


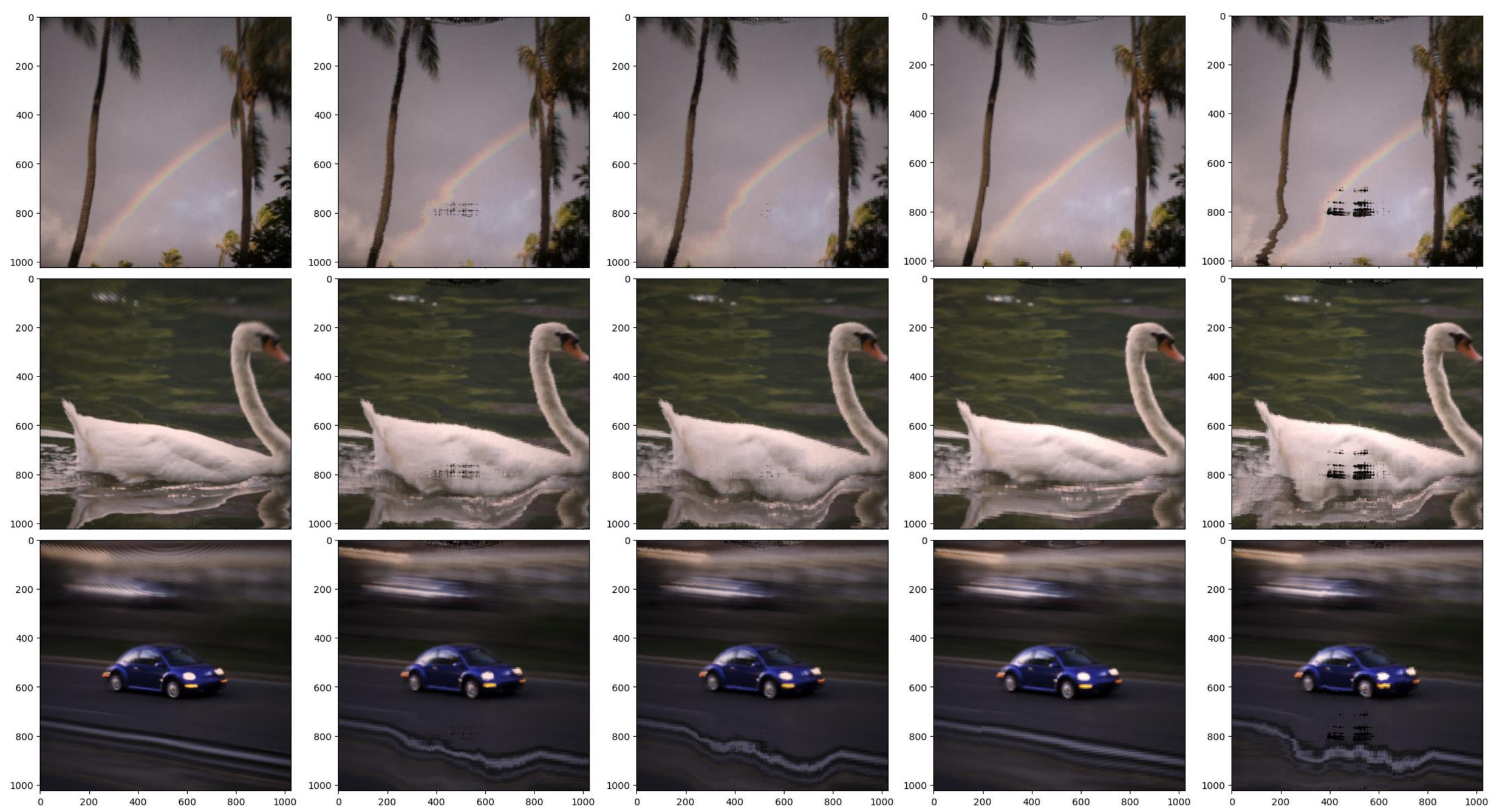




Average PSNR: 22.03 (Std 2.89)







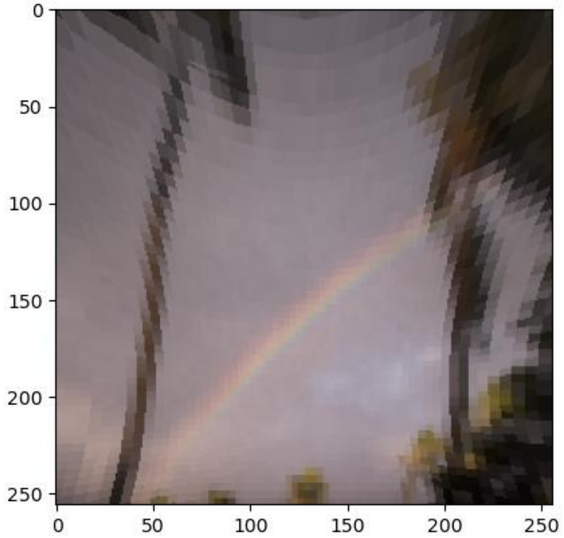
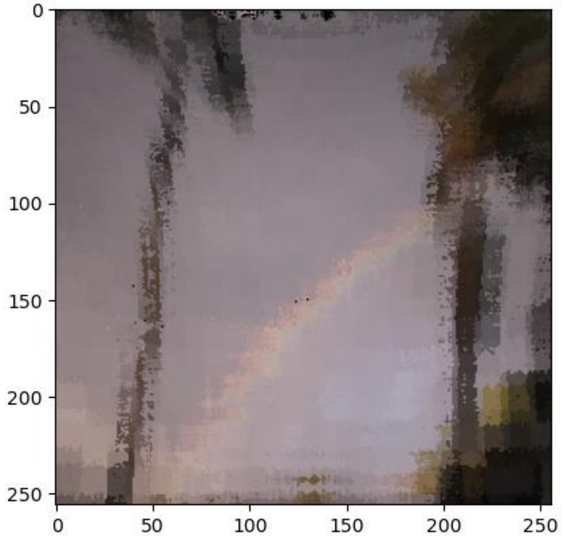
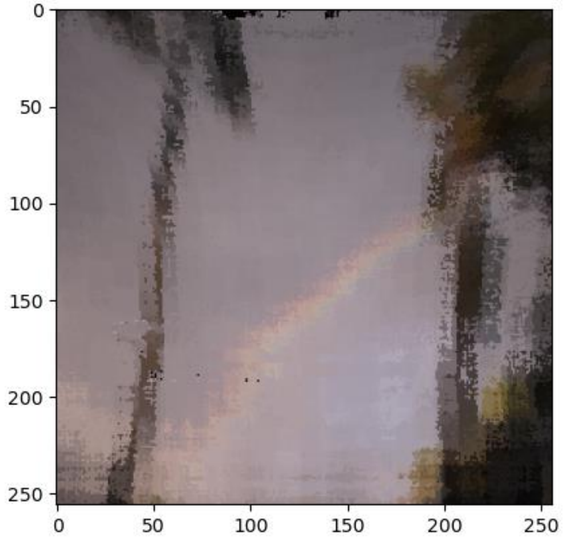
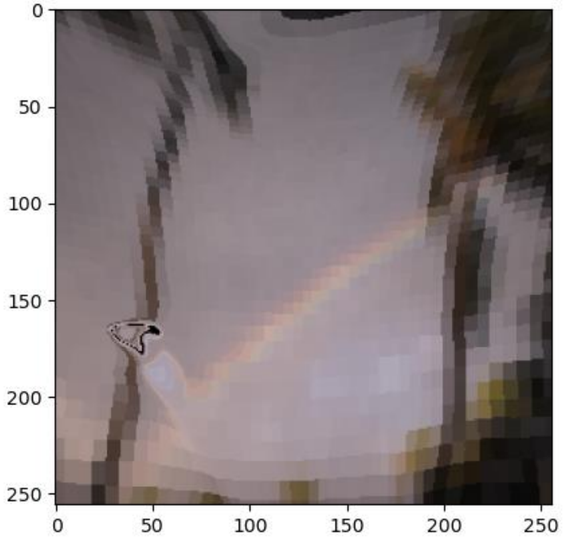
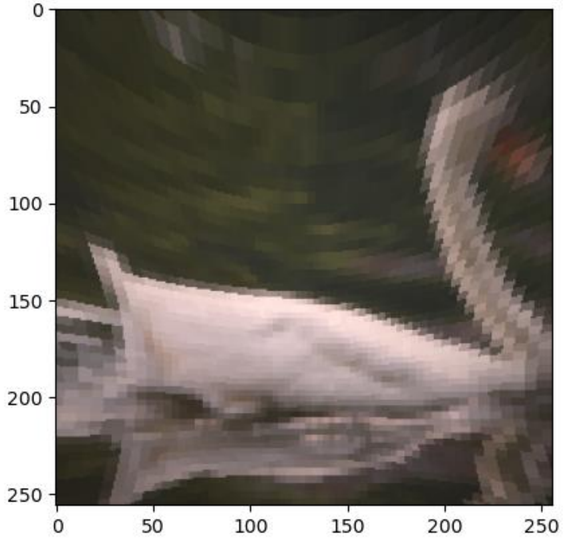
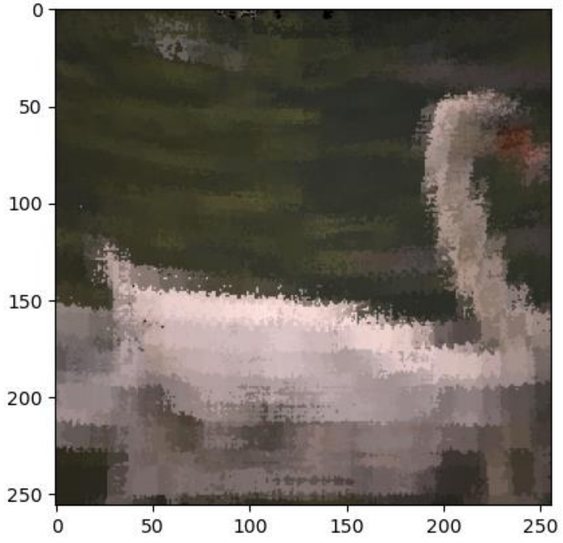
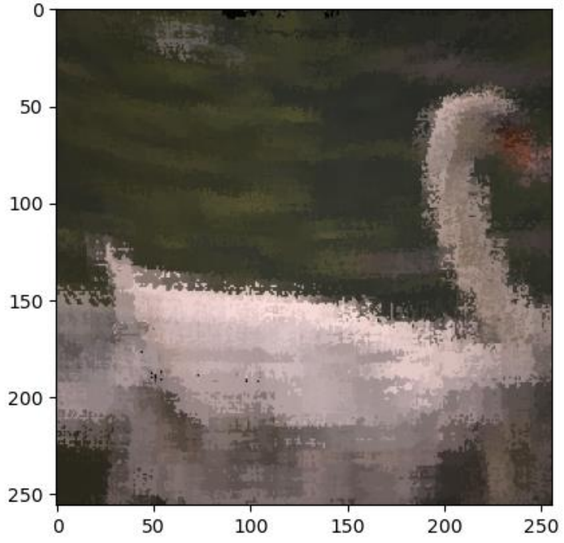
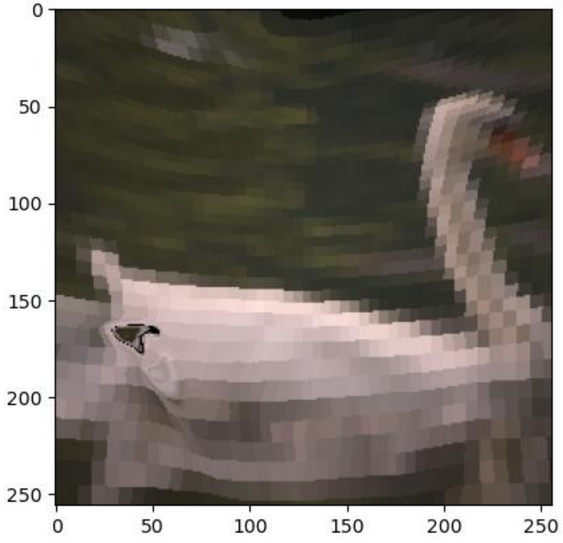
Sphere$_{1024}$*, NE, g=0.80*

Teapot$_{1024}$*, E, g=0.80*

Teapot$_{1024,}$ *E, g=0.30*

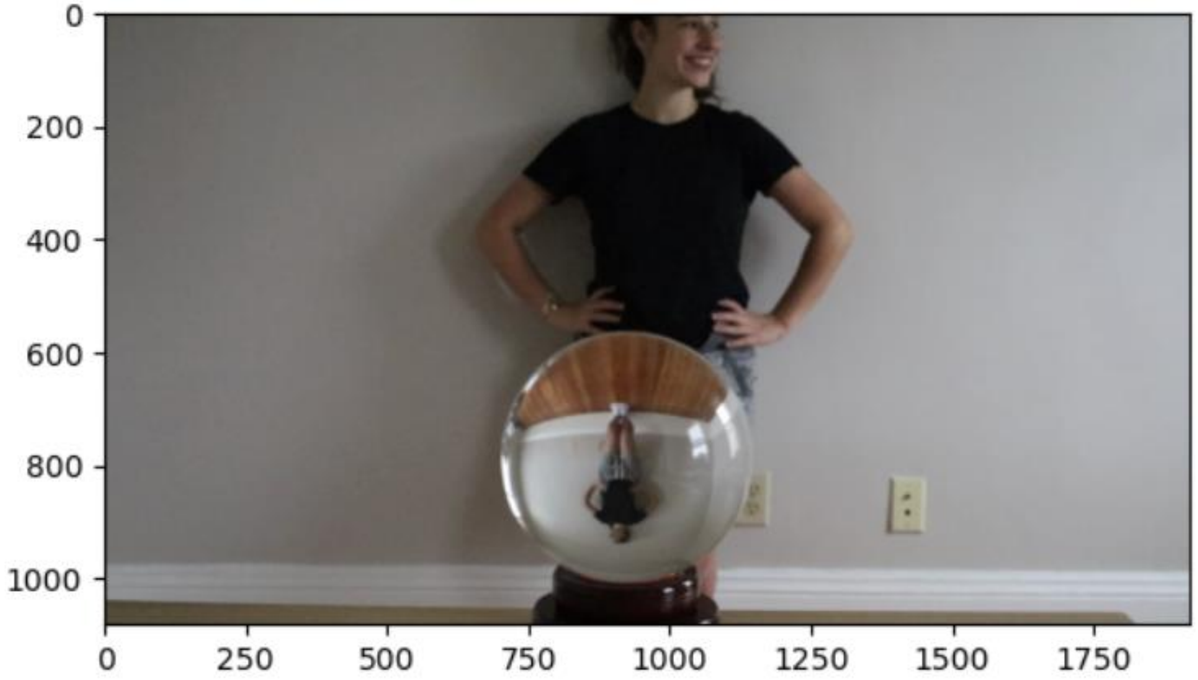
Teapot$_{1024,}$ *NE, g=0.80*

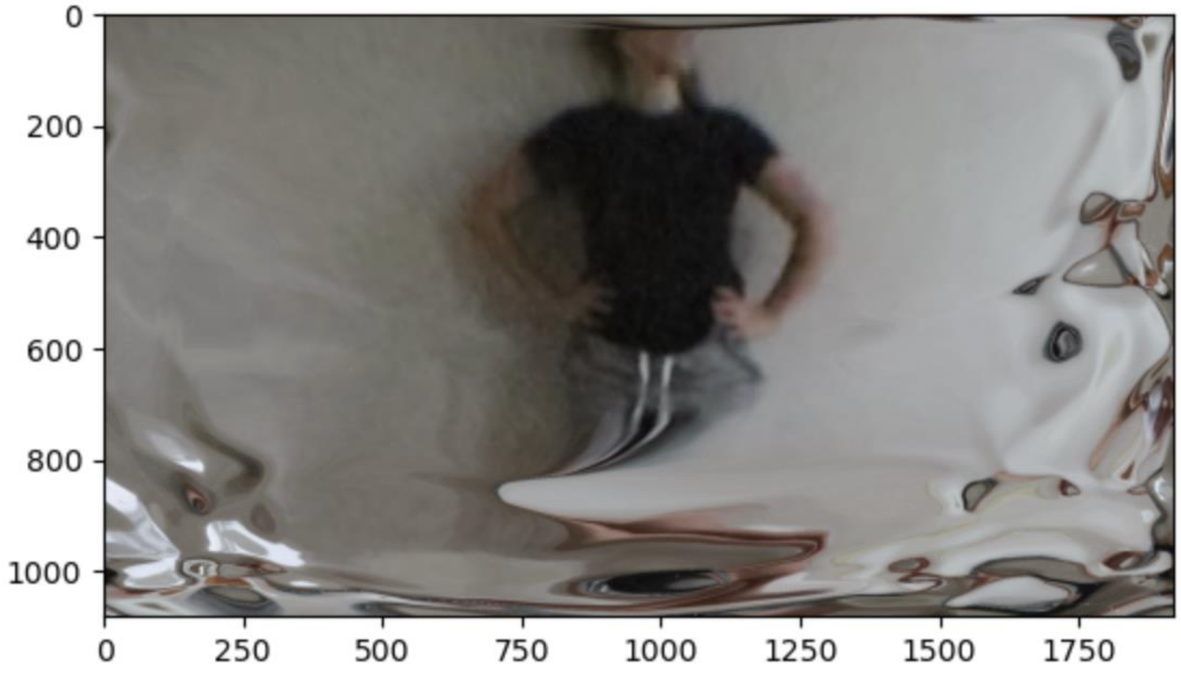
Teapot$_{1024,}$ *MDG*$_{\leq 2}$*,*
E, g=0.80

Sphere$_{256}$, NE	*Teapot$_{256}$, MDG$_{\leq 2}$, E*	*Teapot$_{256}$, MDG$_{Any}$, E*	*Teapot$_{256}$, MDG$_{Any}$, NE, MLP width doubled*
			
PSNR: 18.66	18.96	21.15	19.63
			
PSNR: 16.99	18.01	18.64	19.32

Extra backup (scratch slides)

Real world example





Comparison to Totems reconstructions

Synthetic

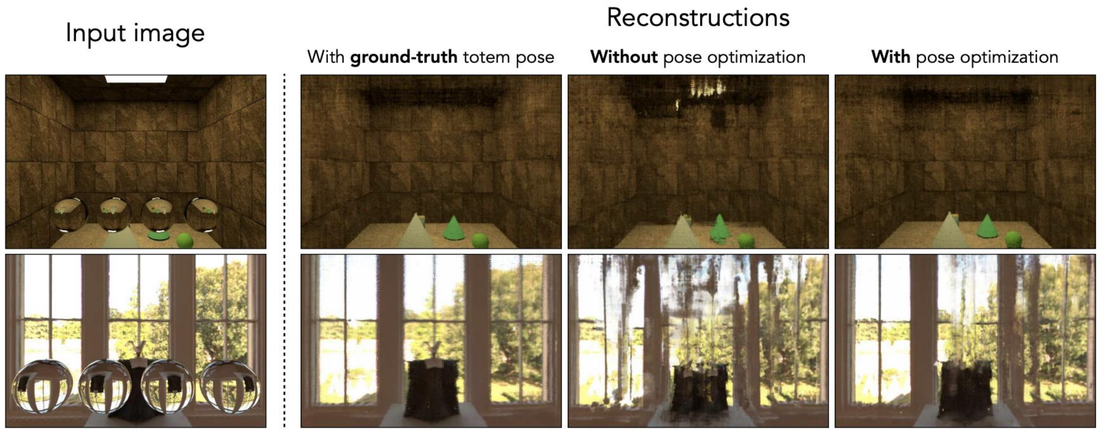
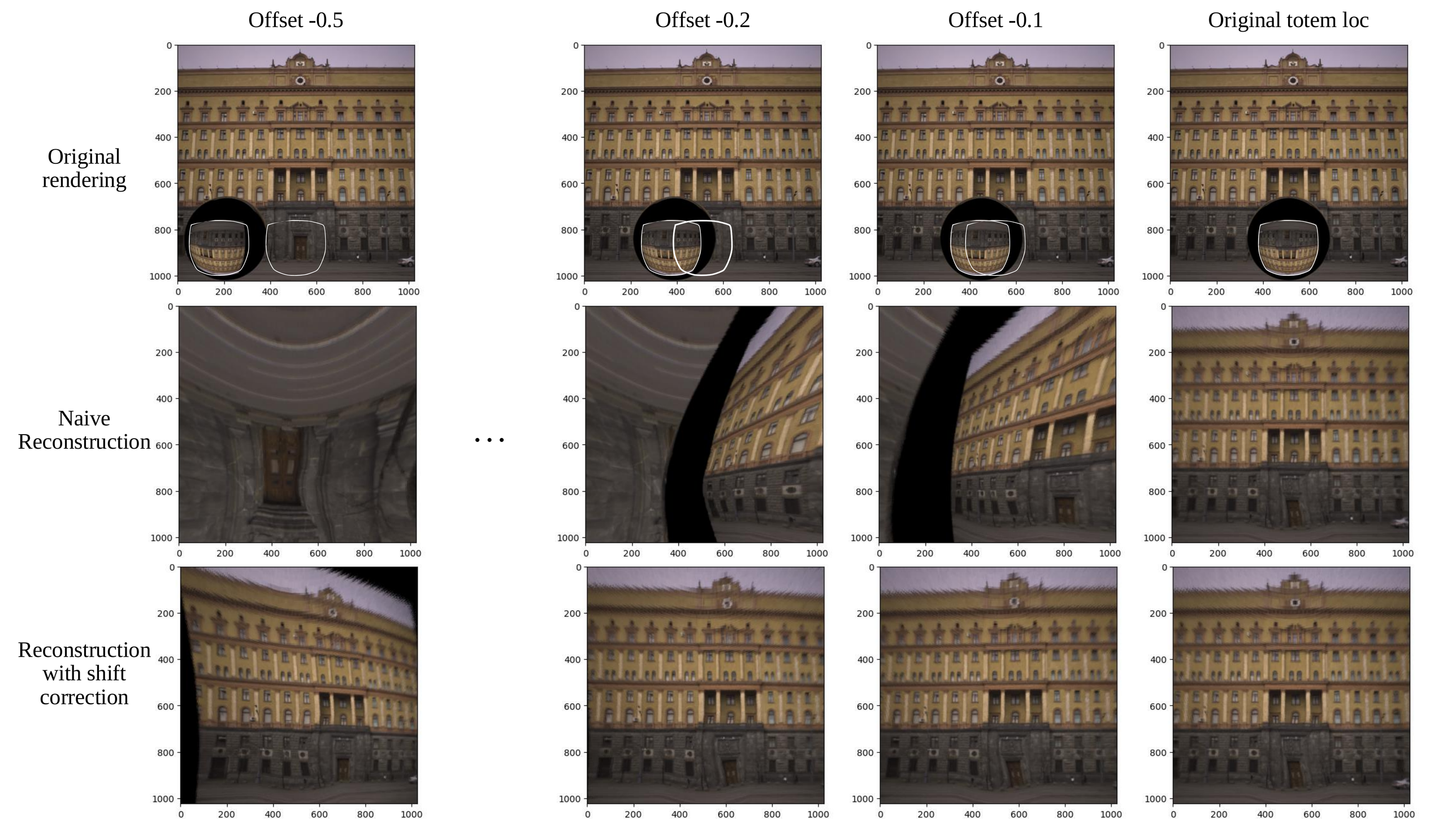


Fig. 5: **Pose optimization on synthetic data:** We start with a setup of spherical totems placed in a simulated environment. We find that four totems is sufficient to recover the scene when the totem pose is accurate; i.e., the ground-truth totem pose used to render the scene. Using the initial estimated totem position leads to artifacts in the reconstructed camera viewpoint, while allowing the totem poses to update while learning the scene representation improves the reconstruction.

Real-world

Fig. 6: **Reconstruction results on real images:** For *real images*, we do not know the ground-truth totem positions and must rely on our position estimates. We find that jointly optimizing the pose of the totems together with the scene reconstruction better recovers the scene geometry and obtains a closer match to the input image than using the initial totem pose estimates. We conduct reconstruction on indoor and outdoor scenes under a range of lighting conditions.


Offset -0.5
Offset -0.2
Offset -0.1
Original totem loc
Original rendering
Naive Reconstruction
Reconstruction with shift correction
• • •