Overview of my research projects

Daniel Stanley Tan

A little bit about me

I come from the Philippines 

Lecturer at De La Salle University (~3 years)

A little bit about me

I come from the Philippines

- Lecturer at De La Salle University (~3 years)

Studied and worked at Taiwan

- PhD at National Taiwan University of Science and Technology

A little bit about me

I come from the Philippines

- Lecturer at De La Salle University (~3 years)

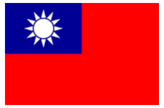
Studied and worked at Taiwan

- PhD at National Taiwan University of Science and Technology
- Research Intern at Inventec

A little bit about me

I come from the Philippines 

- Lecturer at De La Salle University (~3 years)

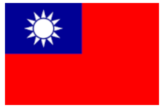
Studied and worked at Taiwan 

- PhD at National Taiwan University of Science and Technology
- Research Intern at Inventec
- Postdoctoral researcher at Academia Sinica

A little bit about me

I come from the Philippines 

- Lecturer at De La Salle University (~3 years)

Studied and worked at Taiwan 

- PhD at National Taiwan University of Science and Technology
- Research Intern at Inventec
- Postdoctoral researcher at Academia Sinica

Now in the Netherlands

- Joining the faculty of Open Universiteit!

A little bit about me

My research interests:

- Computer Vision
- (Deep) Machine Learning
- Creative AI

Anomaly Detection

Defect Detection

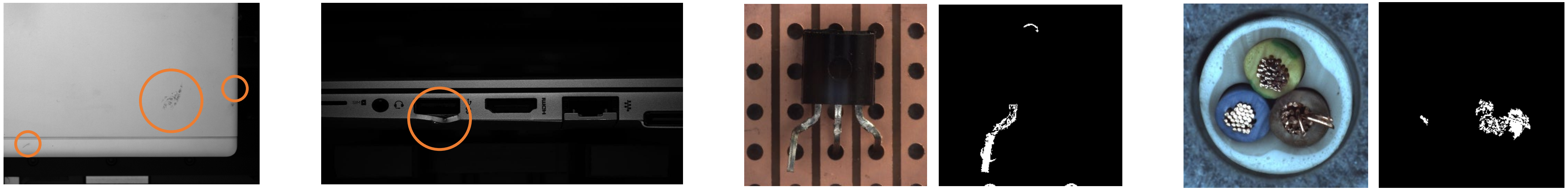
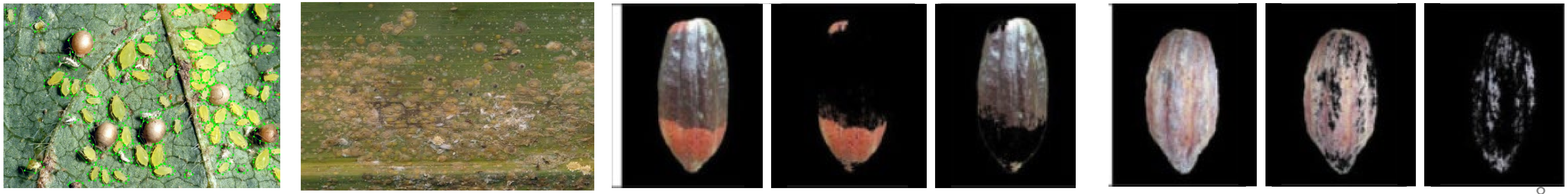


Image Forensics

Crop Pest and Disease Detection



Creative AI

(Controllable) Style Transfer

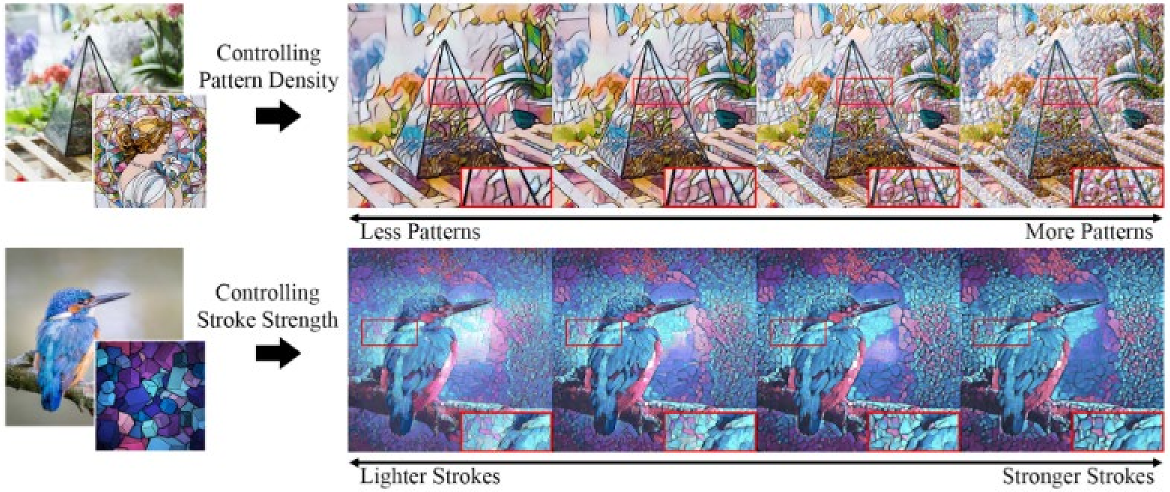


Image-to-image Translation

Defect Detection

Defect Detection

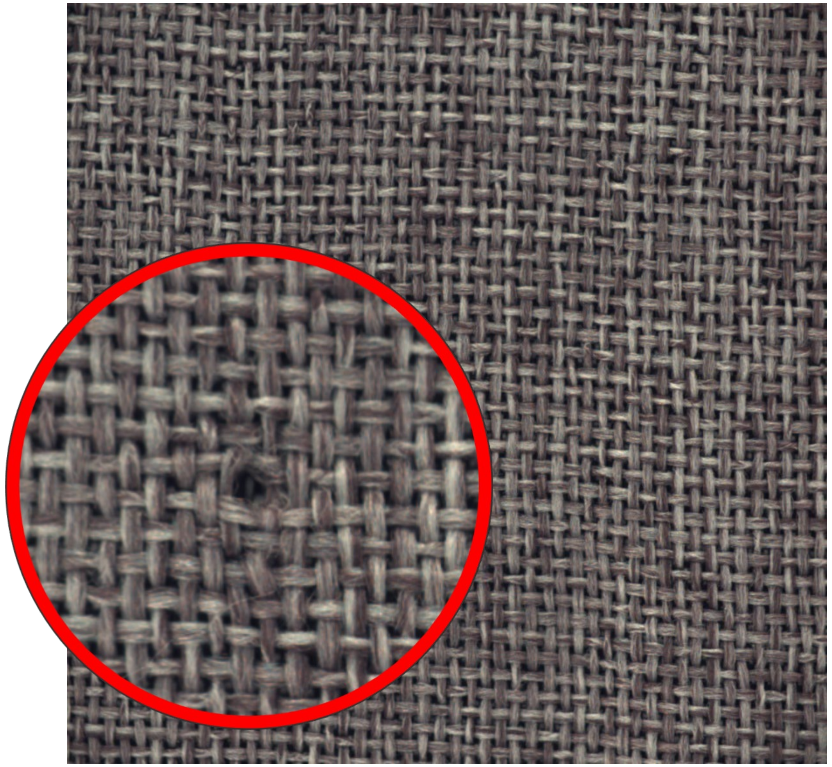
Task of detecting faults or imperfections in a product

Challenge in detecting defects

Differences can be subtle!

Normal

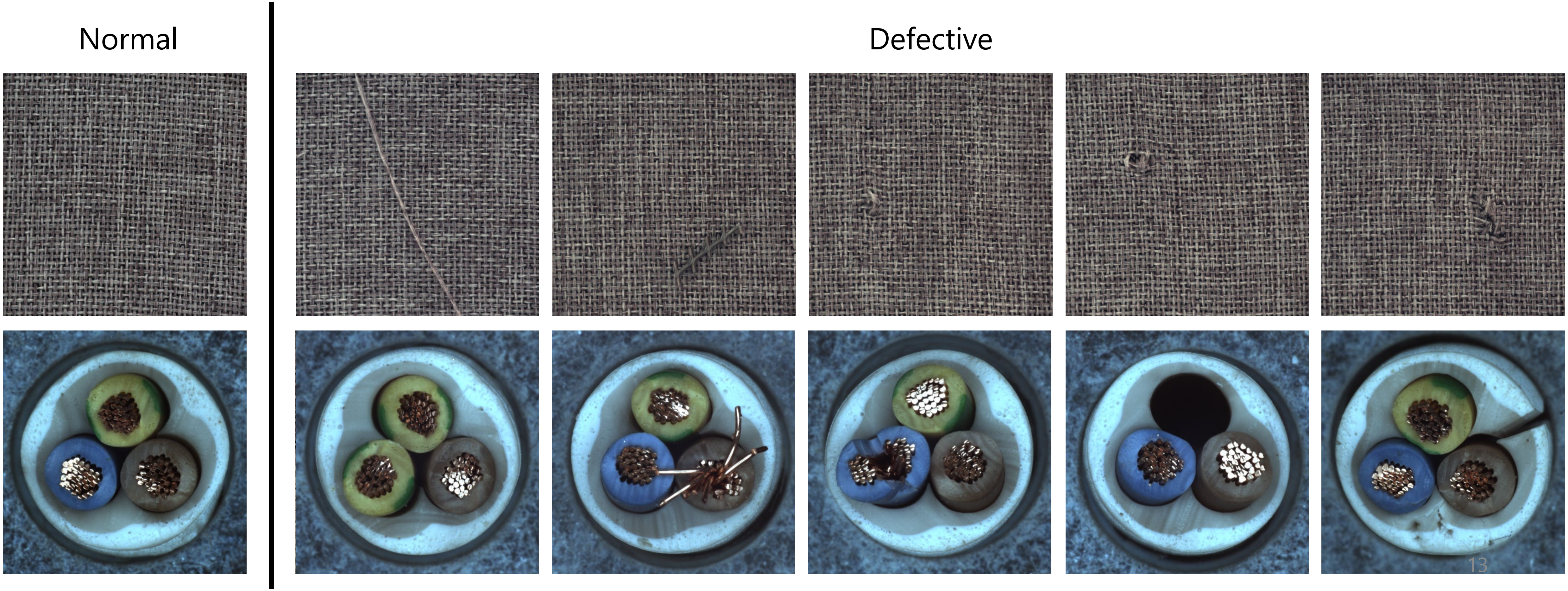
Defective



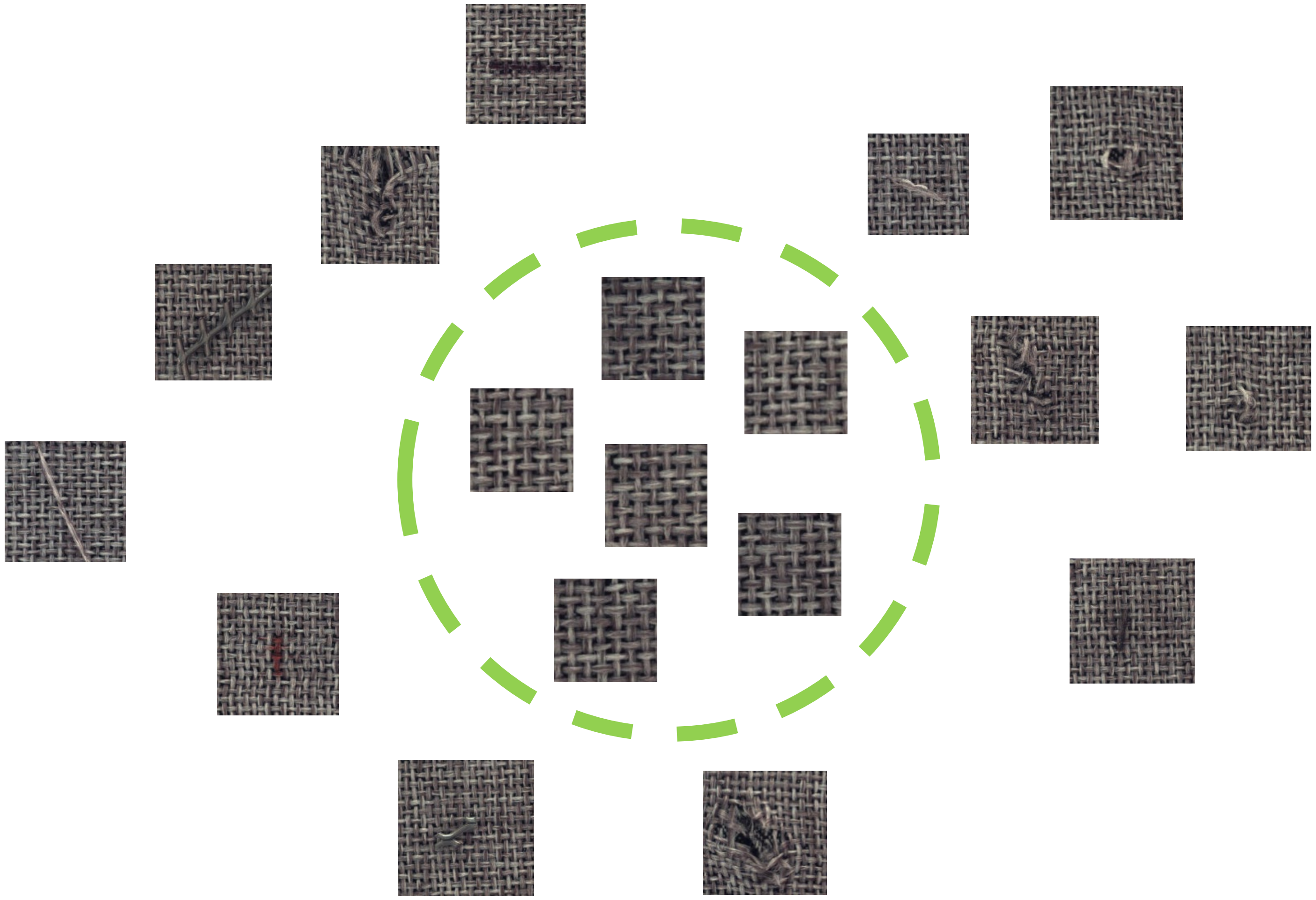
Challenge in detecting defects

Defects can be anything and do not necessarily look alike!

Can't collect a dataset that covers all possible defect types, making it difficult to employ standard classifiers



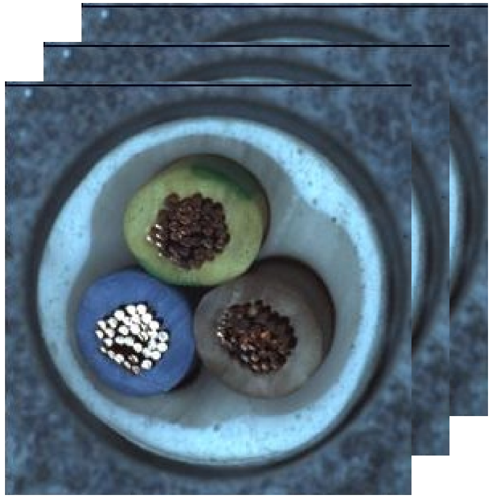
Learn the distribution of normal data



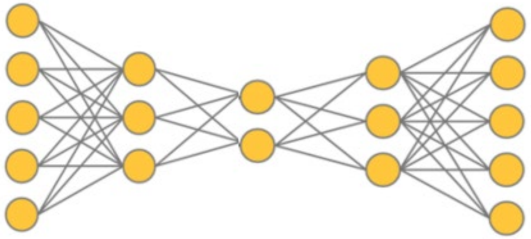
Everything far from normal are considered defects

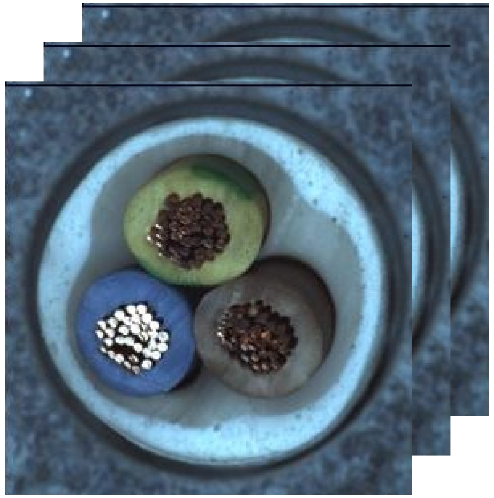
Auto-encoder based defect detection

Training Time



Only Normal Images

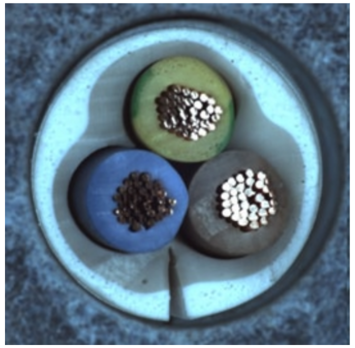




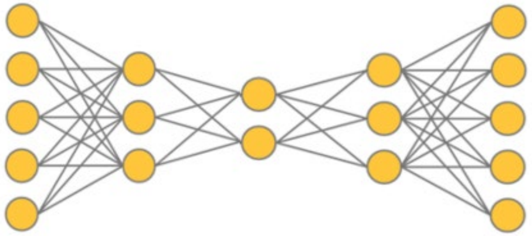
Reconstruction

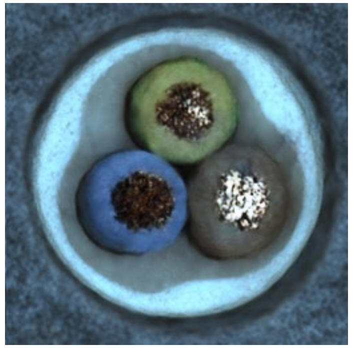
Auto-encoder based defect detection

Test Time

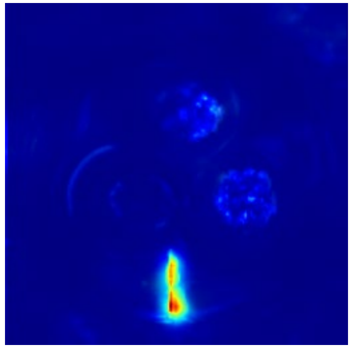


Input Image





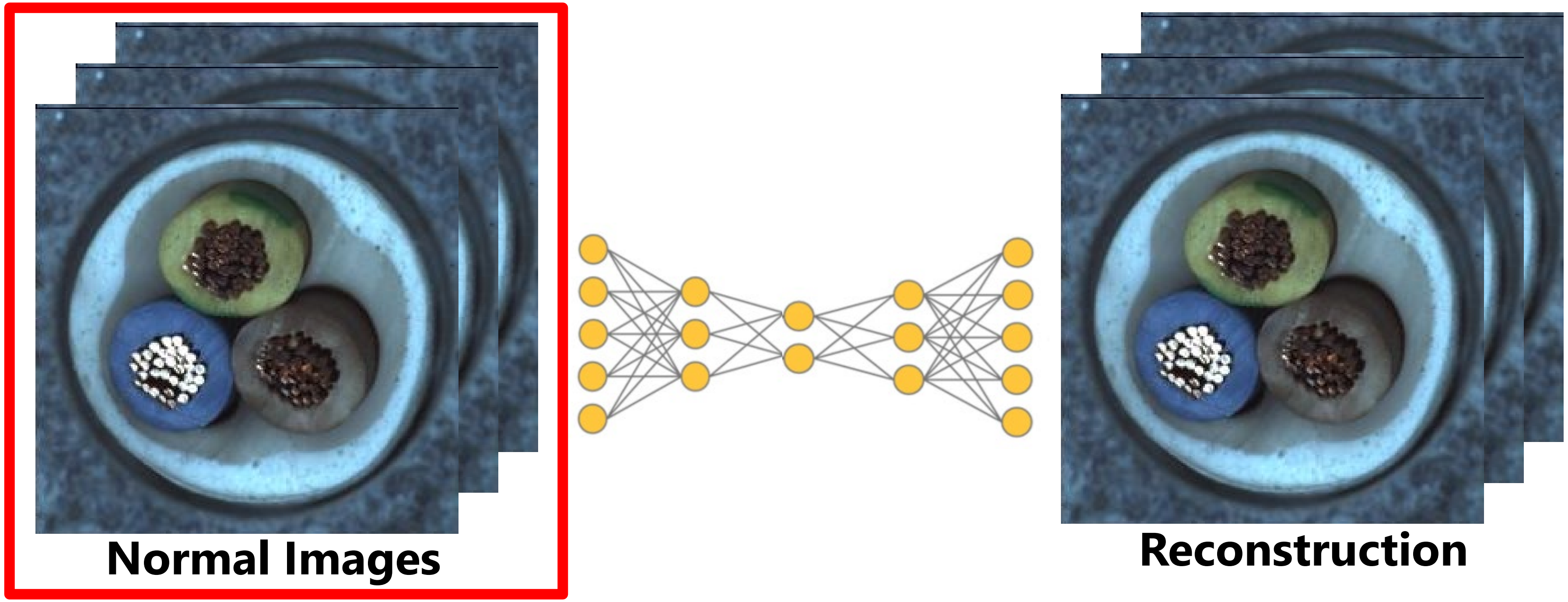
Reconstruction



Difference Map

Limitations

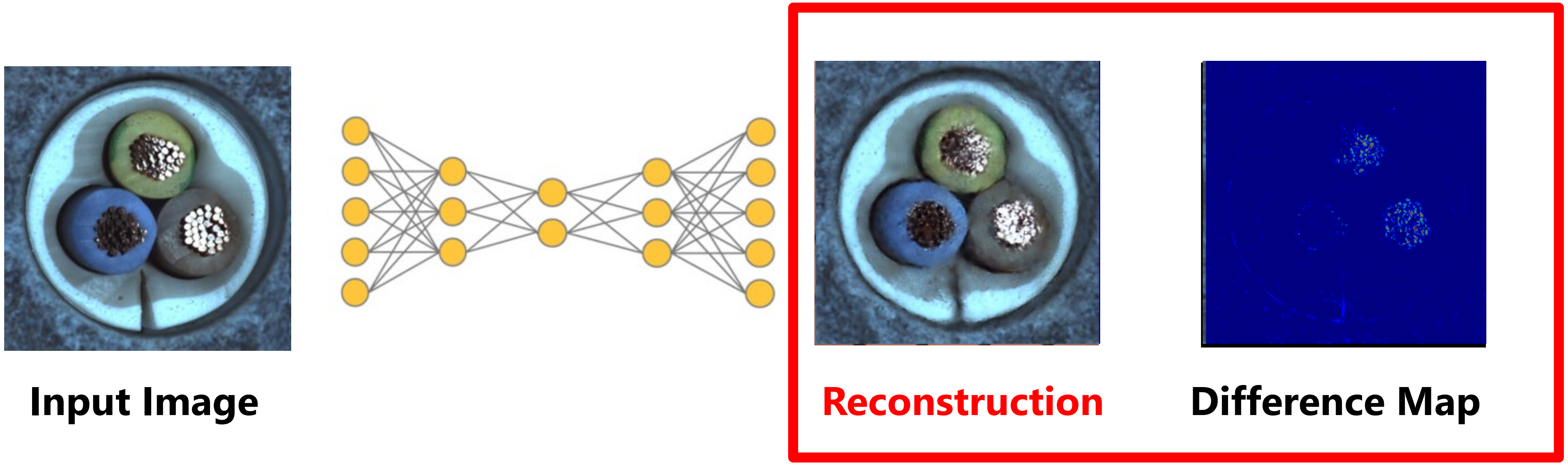
Assumes training data only contains normal images.



Making it difficult for fast changing product designs such as gadgets and laptop models since it adds delays and annotation overhead

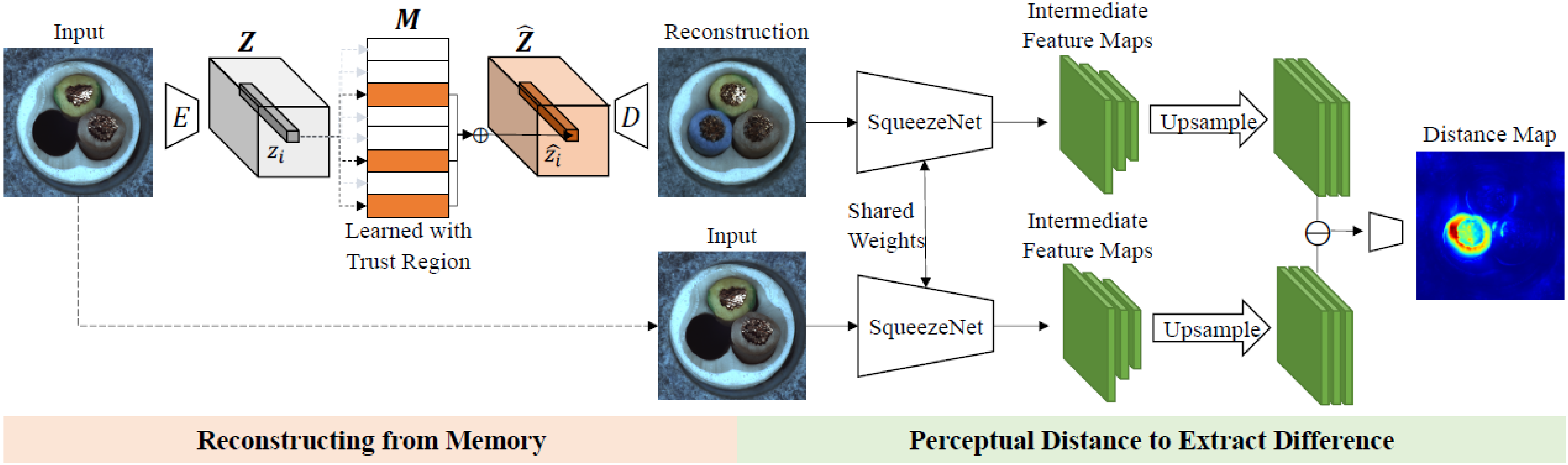
Limitations

Can be overly general and unintentionally reconstruct defects
Further aggravated when noise (defective images) leak into the training data

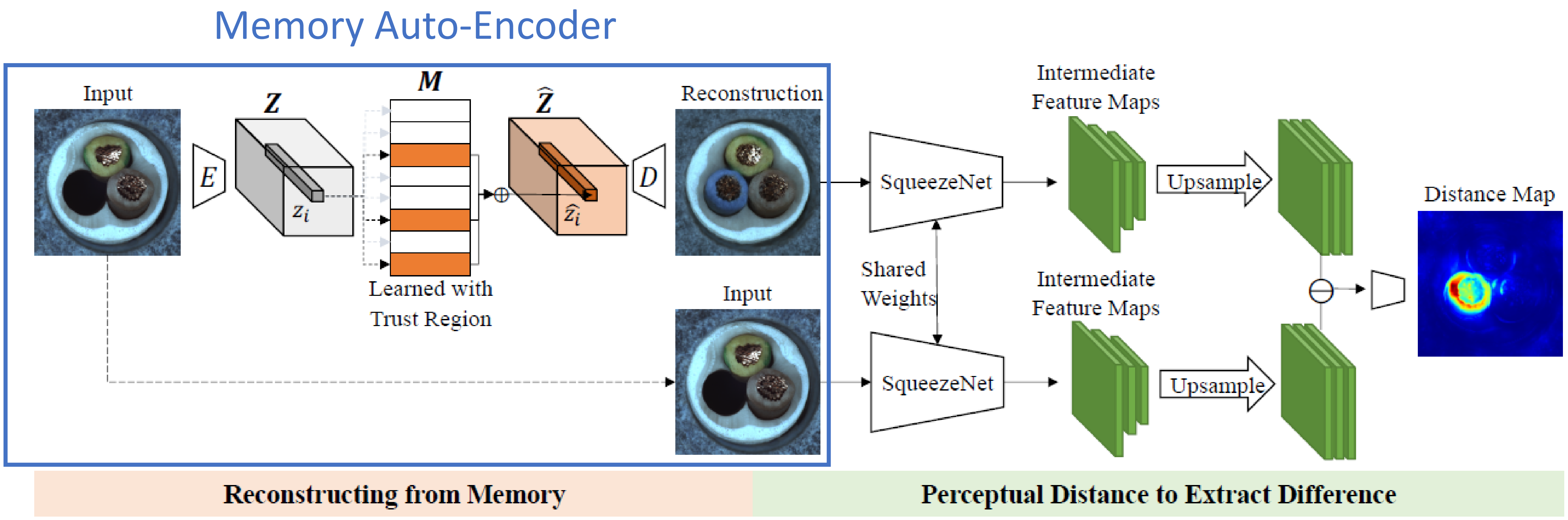


TrustMAE

- Allows training on noisy data, significantly reducing the burden of annotation



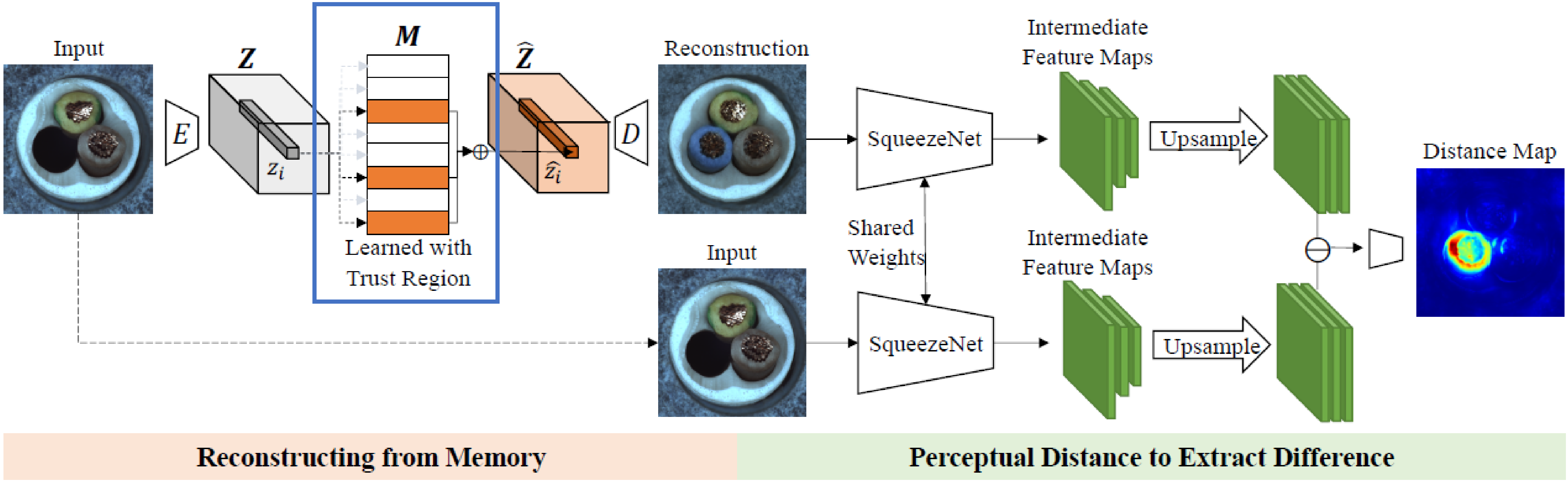
TrustMAE



Reconstructs a normal version of the input.

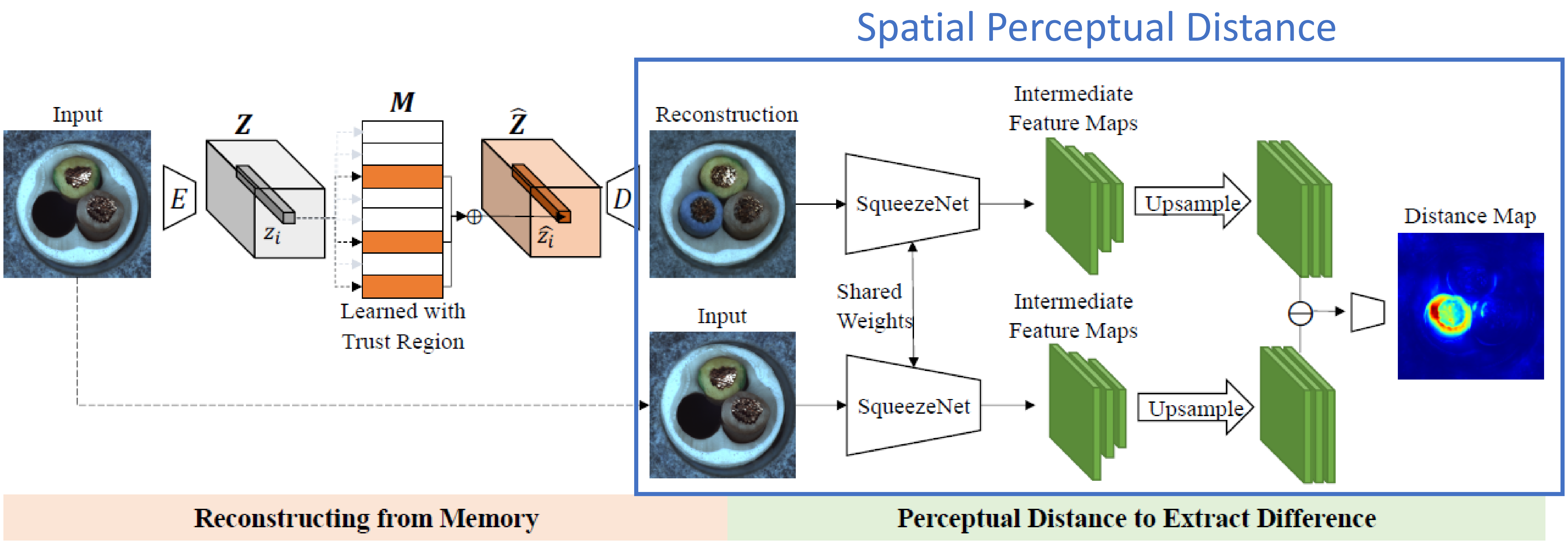
TrustMAE

Trust Region Memory Updates



Prevents memory from being contaminated by defects.

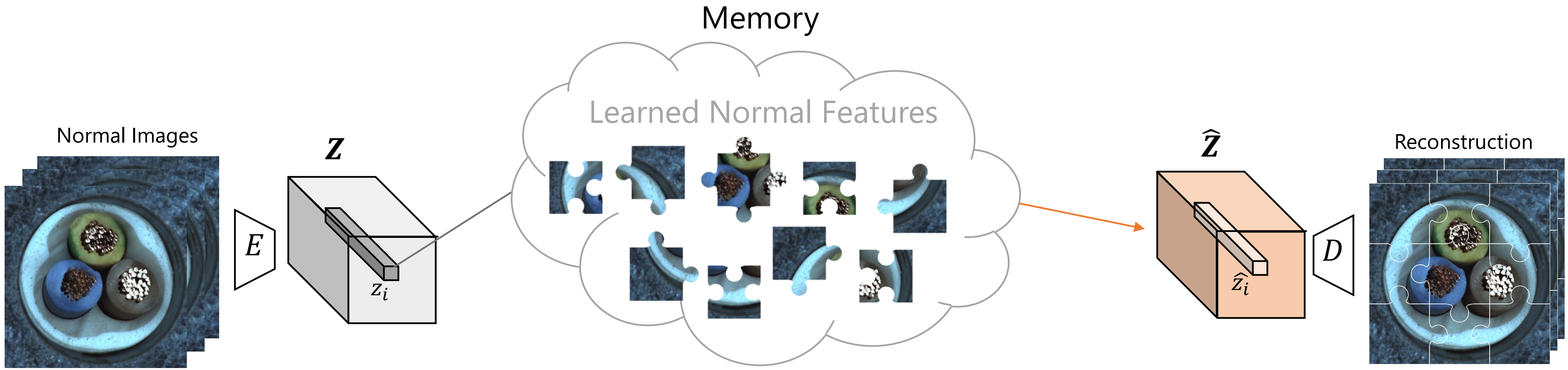
TrustMAE



Computes distance to normal.

Memory Auto-Encoder

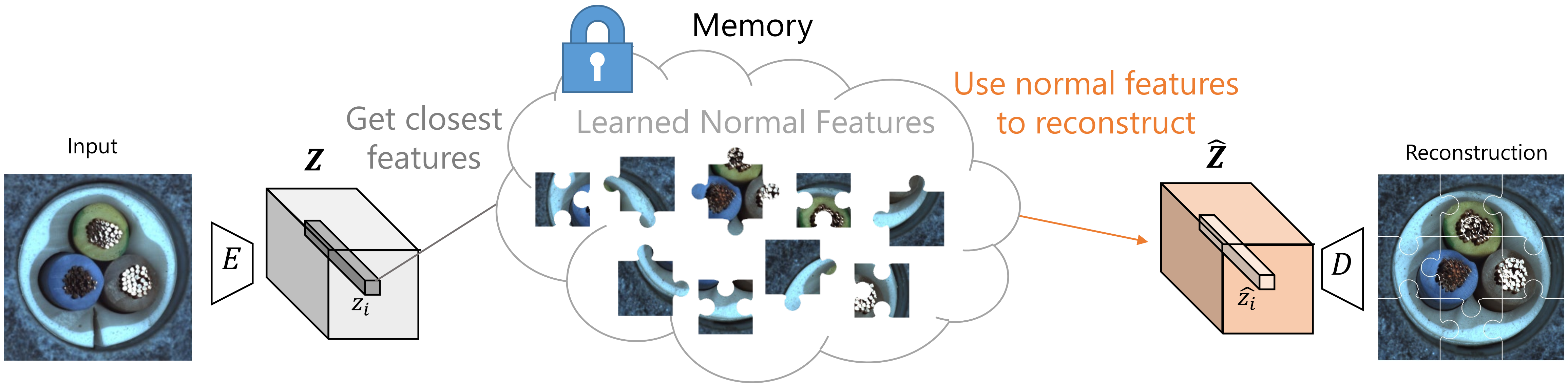
Training Time



*First assume that training contains only normal data. We will remove this constraint later on

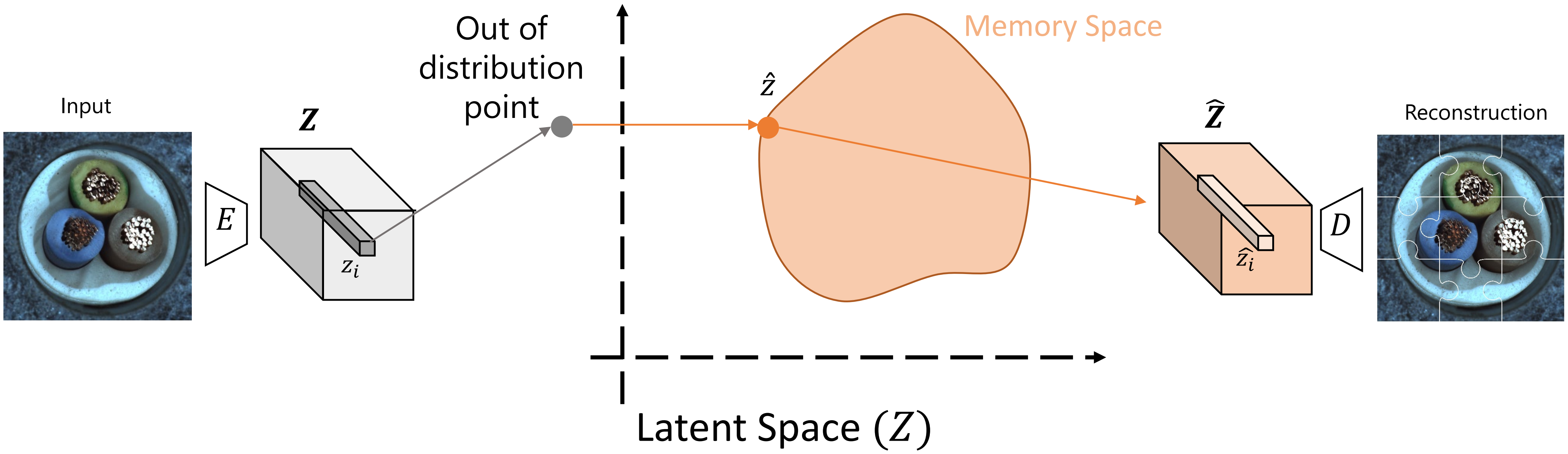
Memory Auto-Encoder

Test Time



Memory Auto-Encoder

With Memory

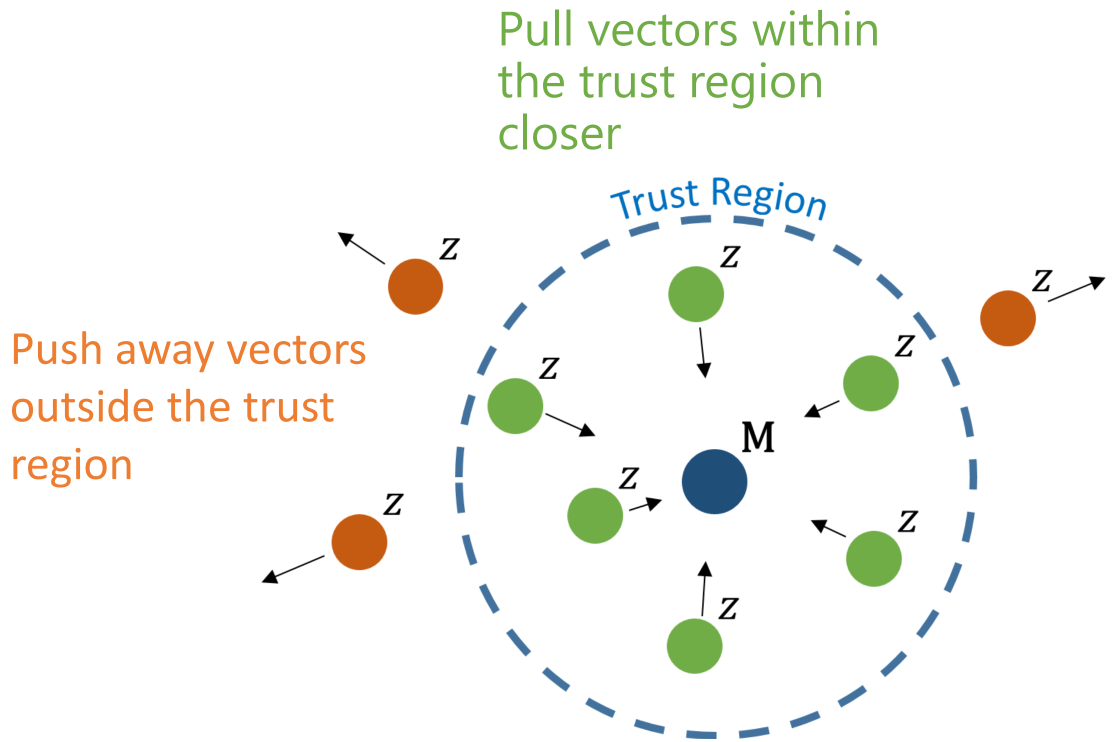


Since we are projecting the point to the memory space, we will always construct normal images

Memory Auto-Encoder

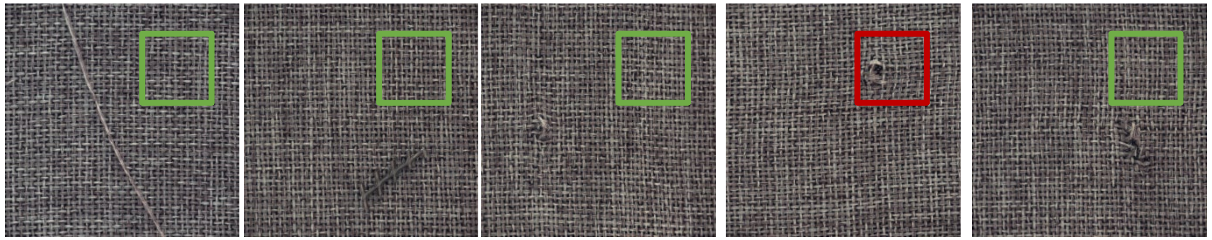
Problem: Given noisy data, how do we ensure the memory space is clean (i.e. defect-free)?

Trust Region Memory Updates



Two key assumptions:

- Defects do not always appear in the same location.

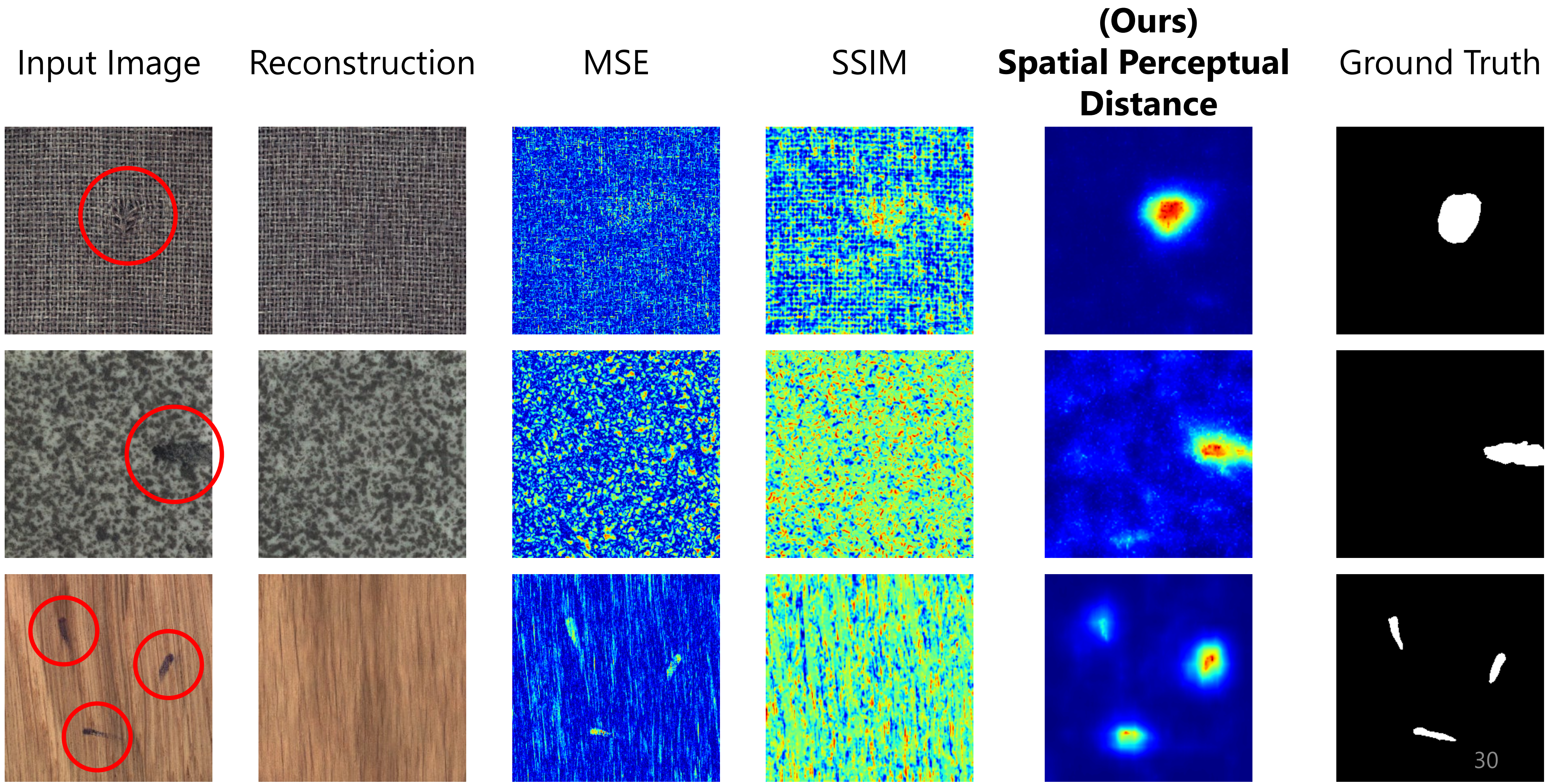


- Normal data have regularity in appearance

Now we have a noise resilient memory auto-encoder.

We need to compute the input’s distance to the reconstructed normal

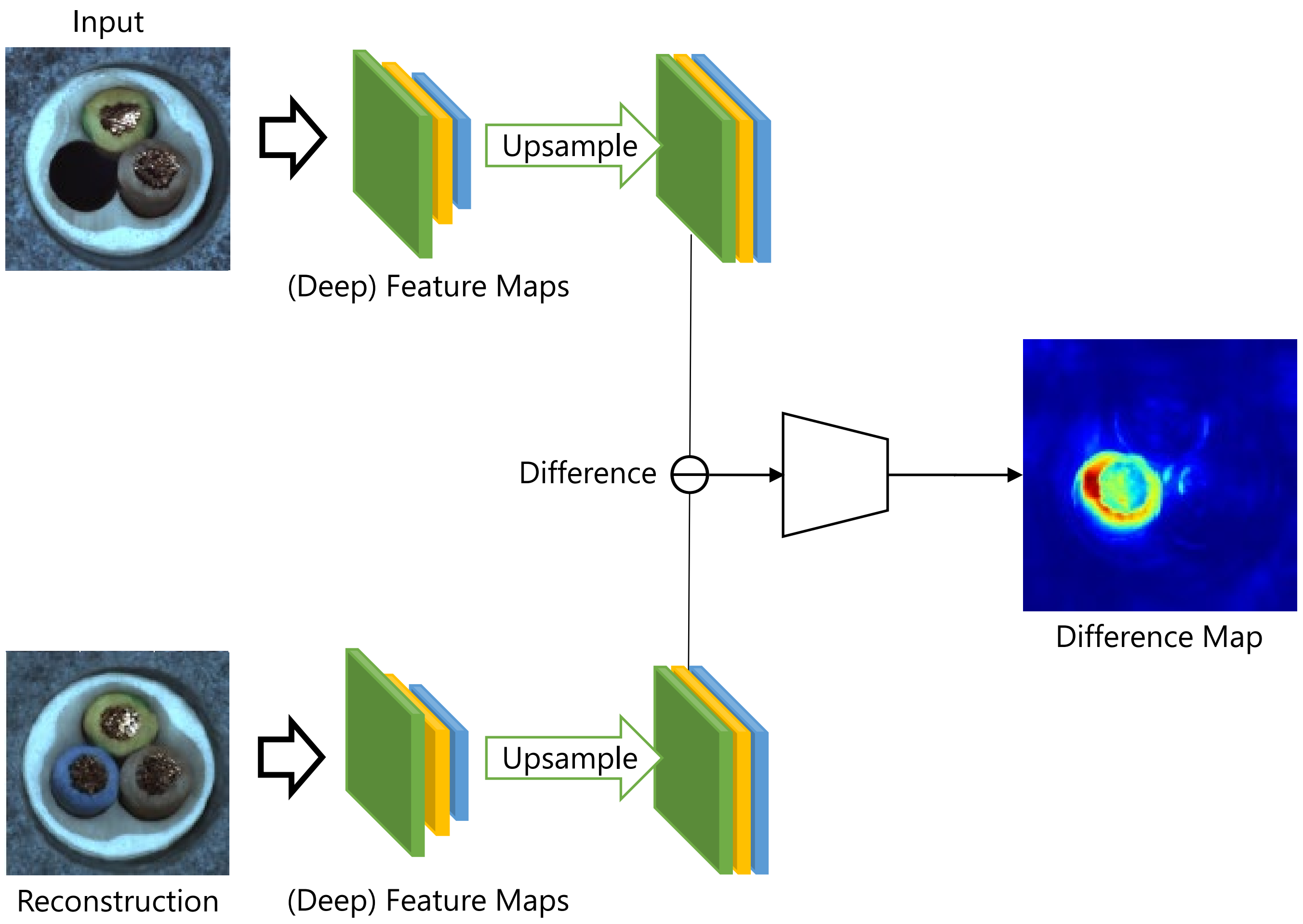
Shallow distances are not enough



Spatial Perceptual Distance

- Captures texture and high level features extracted by the network in computing distances

- Contains invariances learned by the network



Zhang et al. CVPR '18

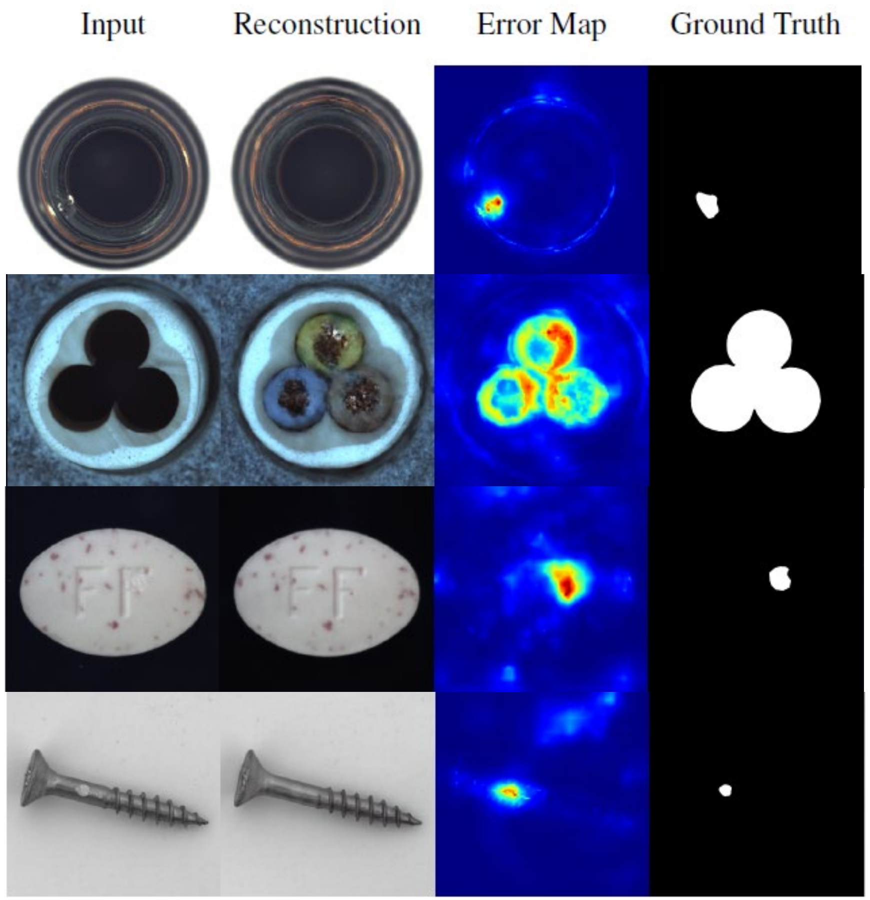
Classification Performance (Image-level AUC)

Method	mean AUC
GeoTrans [1]	67.23
GANomaly [2]	76.15
ARNet [3]	83.93
f-Ano-GAN [4]	65.85
MemAE [5]	81.85
TrustMAE-noise free	**90.78**

Segmentation Performance (Pixel-level AUC)

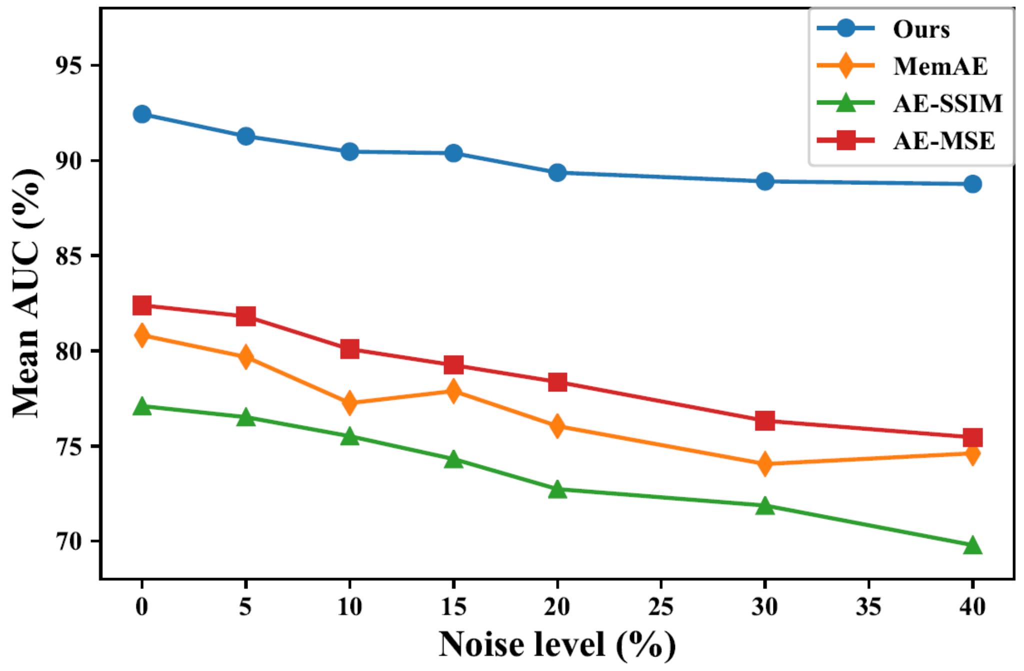
Method	mean AUC
AE-L2 [6]	80.40
AE-SSIM [6]	81.83
MemAE [5]	85.74
Towards Visually Explaining [7]	86.07
CNN Feature Dictionary [8]	78.07
AnoGAN [9]	74.27
AE-SSIM Grad [10]	86.38
γ-VAE Grad [10]	88.77
AE-L2 Grad [10]	88.77
VAE Grad [10]	*89.29*
TrustMAE-noise free	**93.94**

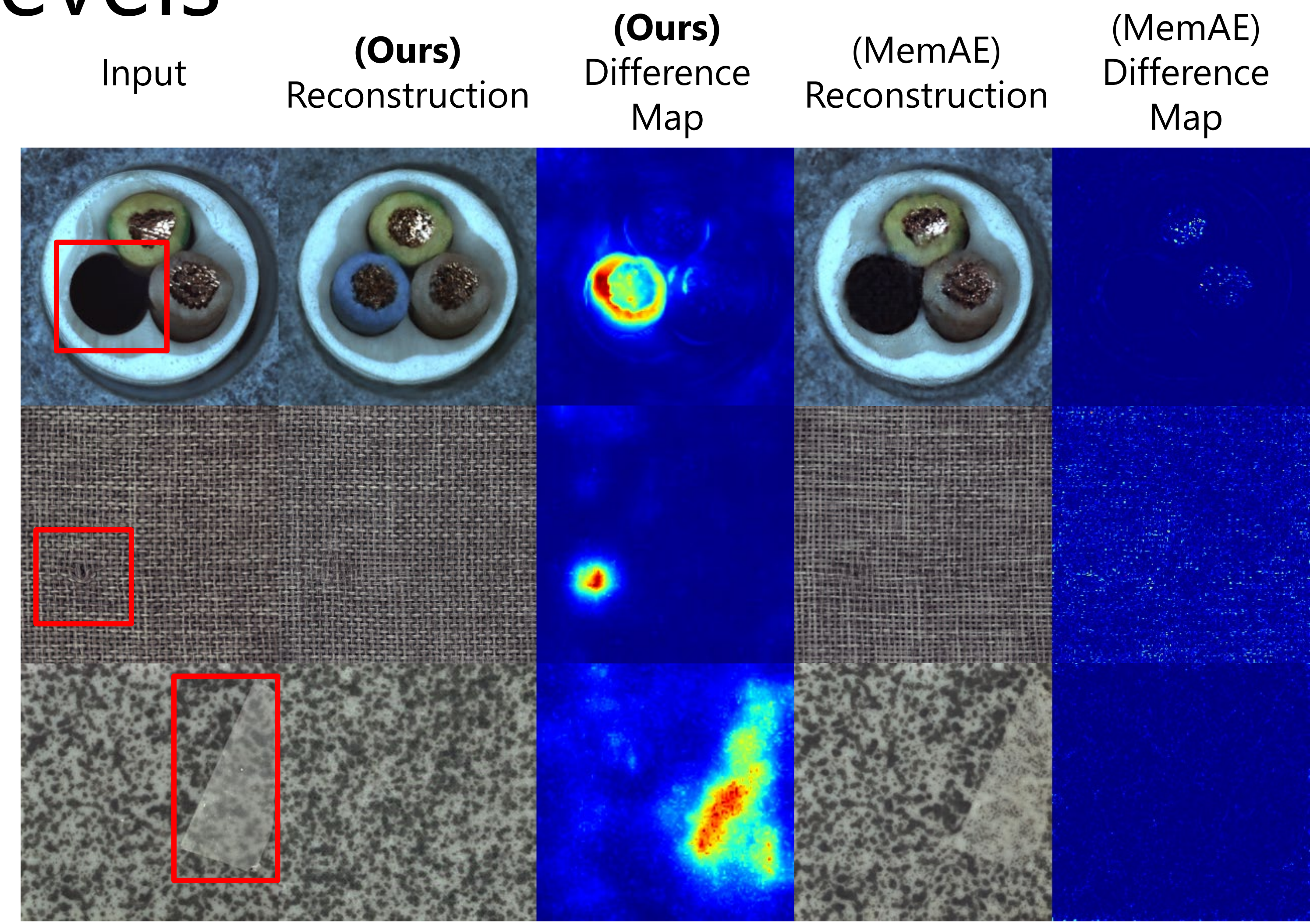
Visual Results



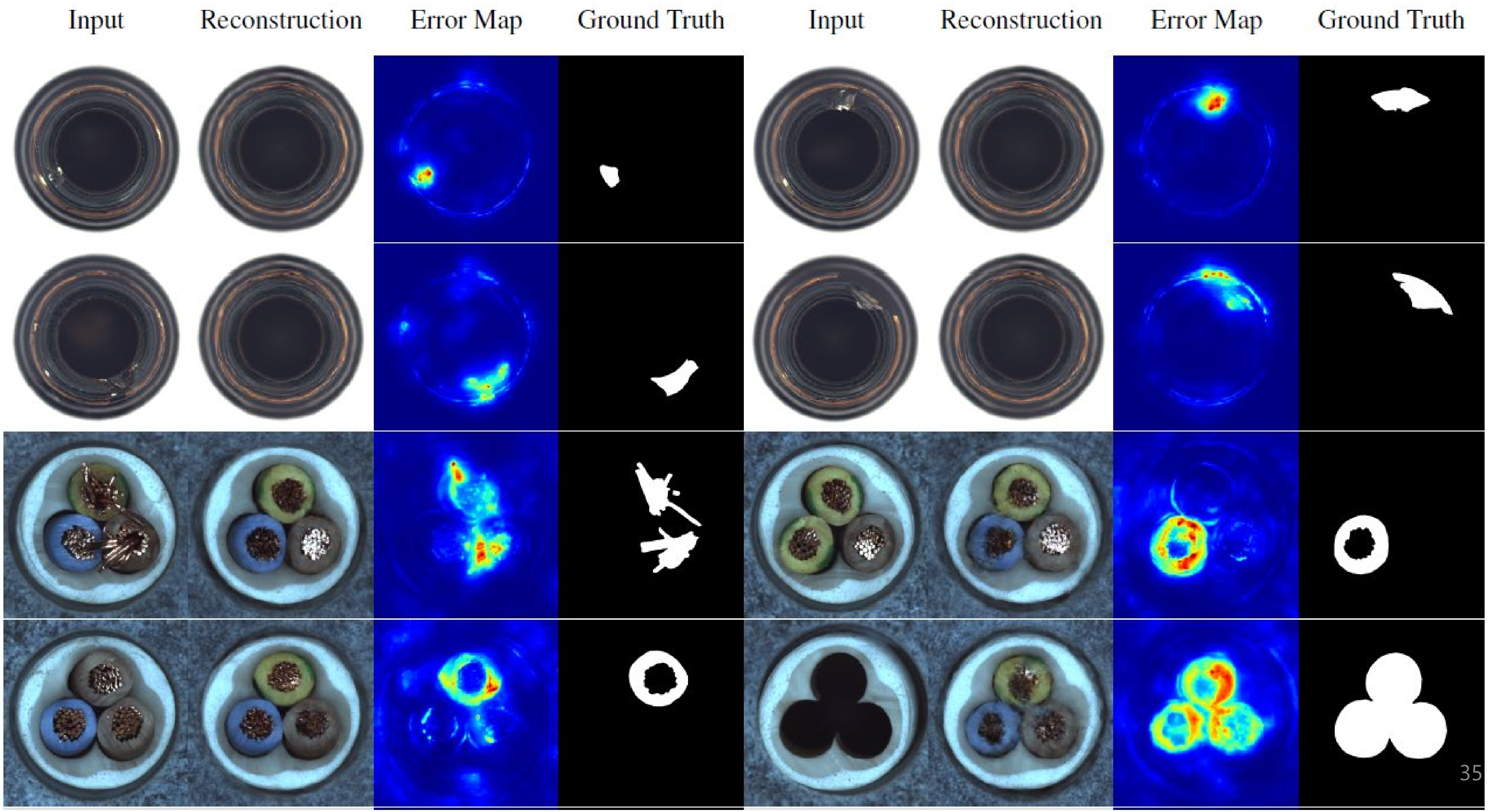
[1] [NeurIPS '18] Golan et al. Deep anomaly detection using geometric transformations.
[2] [ACCV'18] Akcay et al. Ganomaly: Semi-supervised anomaly detection via adversarial training.
[3] [arxiv'20] Huang et al. Inverse-transform autoencoder for anomaly detection
[4] [Medical image analysis '19] Schlegl et al. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks
[5] [ICCV'19] Gong et al. Memorizing Normality to detect anomaly.
[6] [VISIGRAPP '19] Bergmann et al. Improving Unsupervised Defect Segmentation by Applying Structural Similarity to Autoencoders
[7] [CVPR'20] Liu et al. Towards visually explaining variational autoencoders.
[8] [Sensors '19] Napoletano et al. Anomaly detection in nanofibrous materials by cnn-based self-similarity.
[9] [CIRP '19] Staar et al. Anomaly detection with convolutional neural networks for industrial surface inspection.
[10] [ICLR '20] Dehaene et al. Iterative energy-based projection on a normal data manifold for anomaly localization

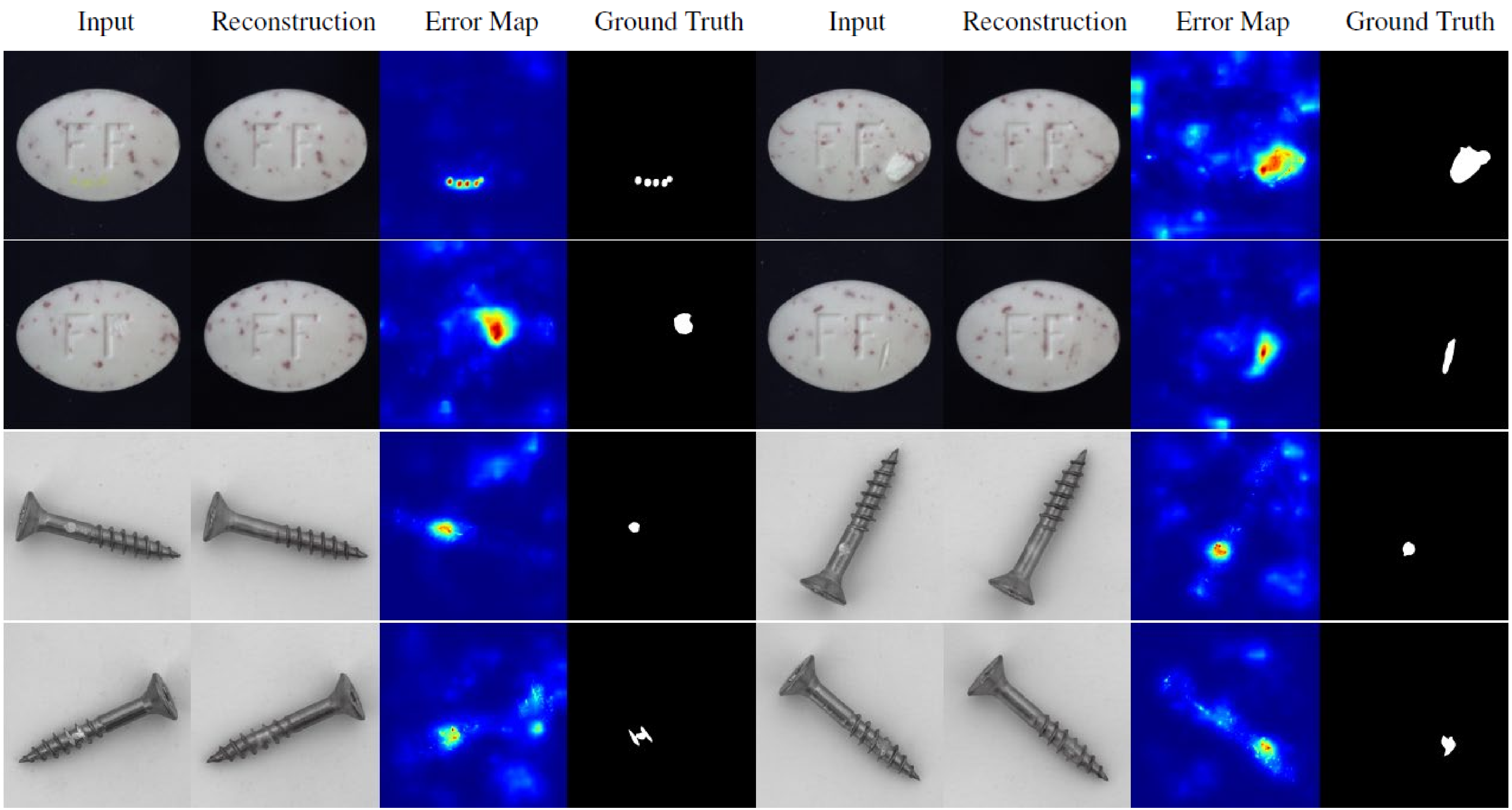
Different Noise Levels





Visual Results




Input
Reconstruction
Error Map
Ground Truth
Input
Reconstruction
Error Map
Ground Truth

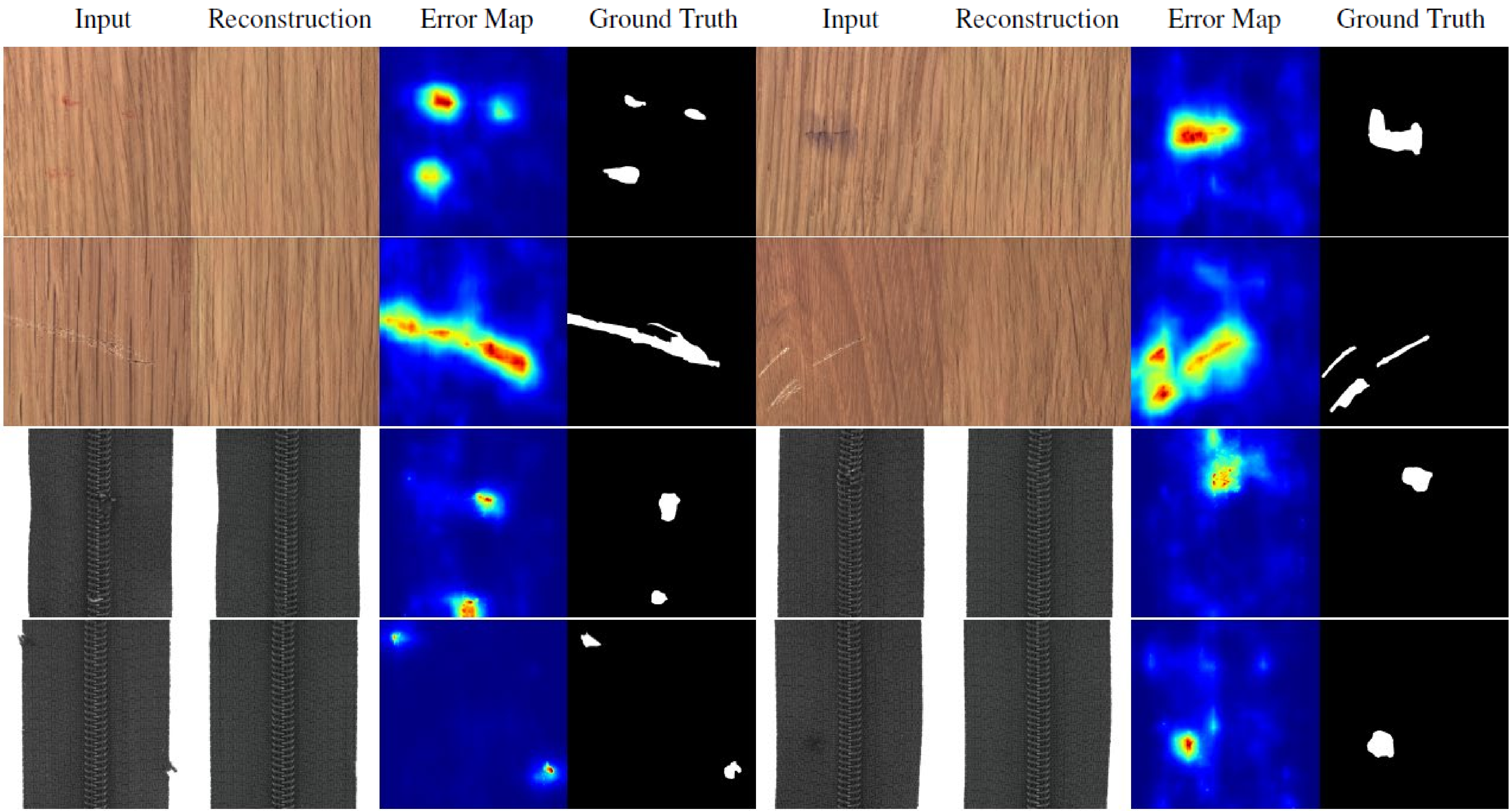

Input
Reconstruction
Error Map
Ground Truth
Input
Reconstruction
Error Map
Ground Truth

Image Forensics

Pristine

Fake

Pristine

Fake

Challenges

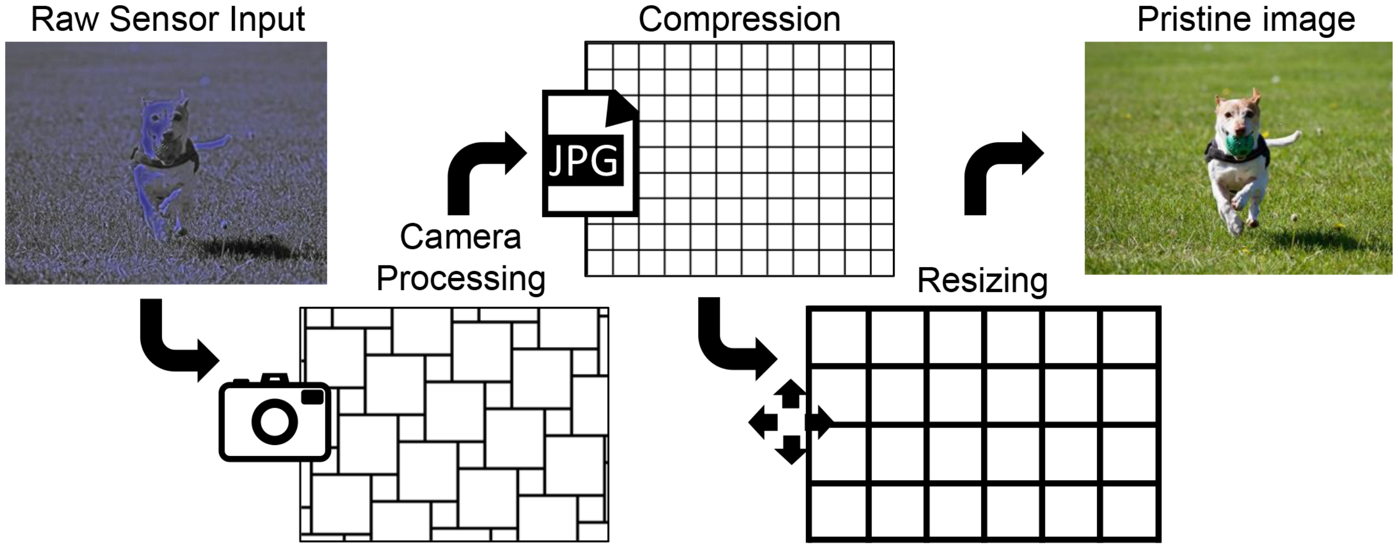
Not easily perceptible

- A good fake image hides its manipulations cleverly with the semantic contents of the image

Hard to extract and isolate weak signals

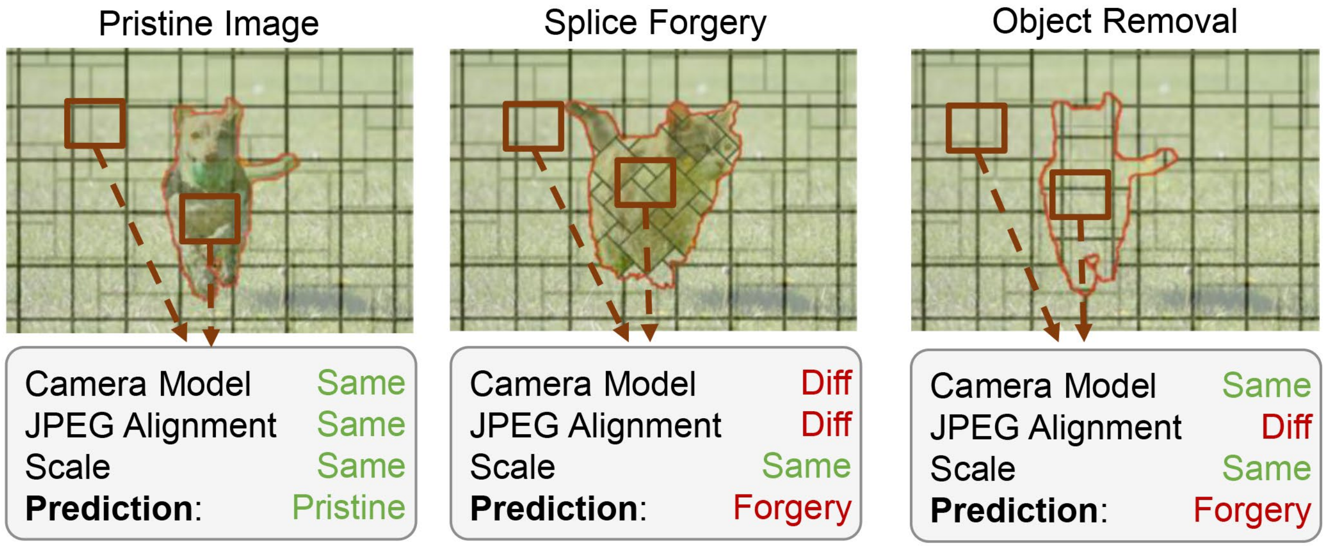
Main Idea

An image undergoes several stages of processing, each of which imprints a spatial signature onto the image.



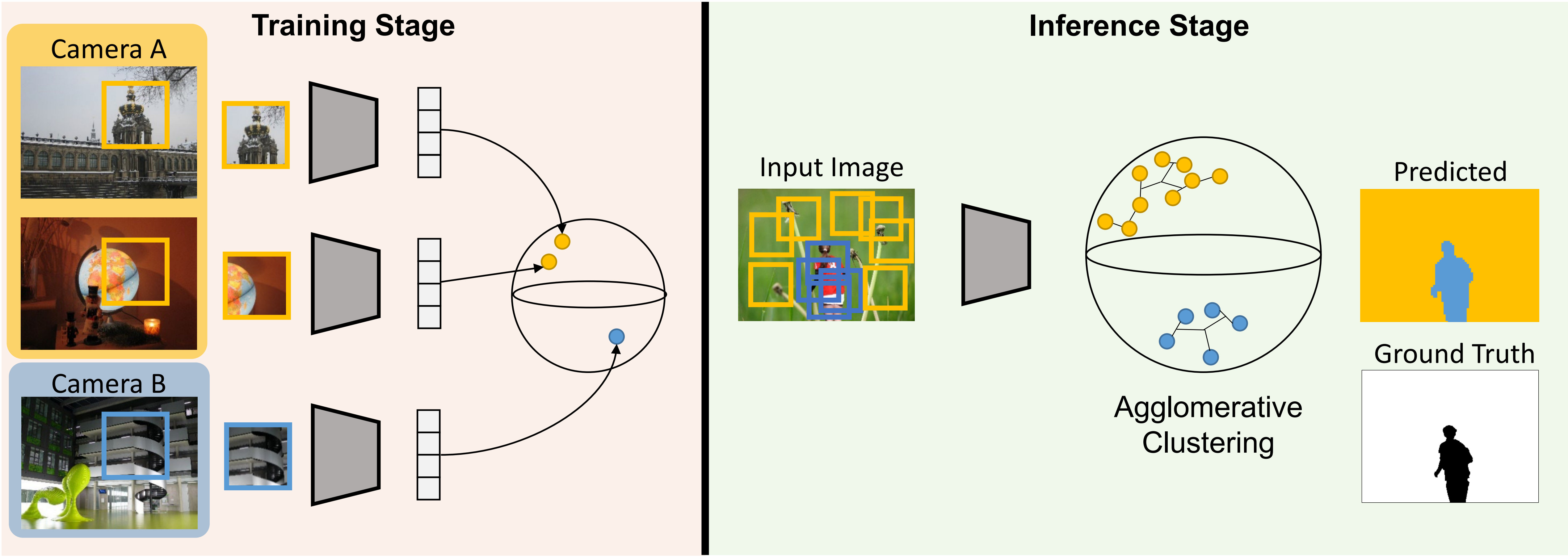
Main Idea

Under pristine conditions, these signatures are regular, but for forgeries these are broken.

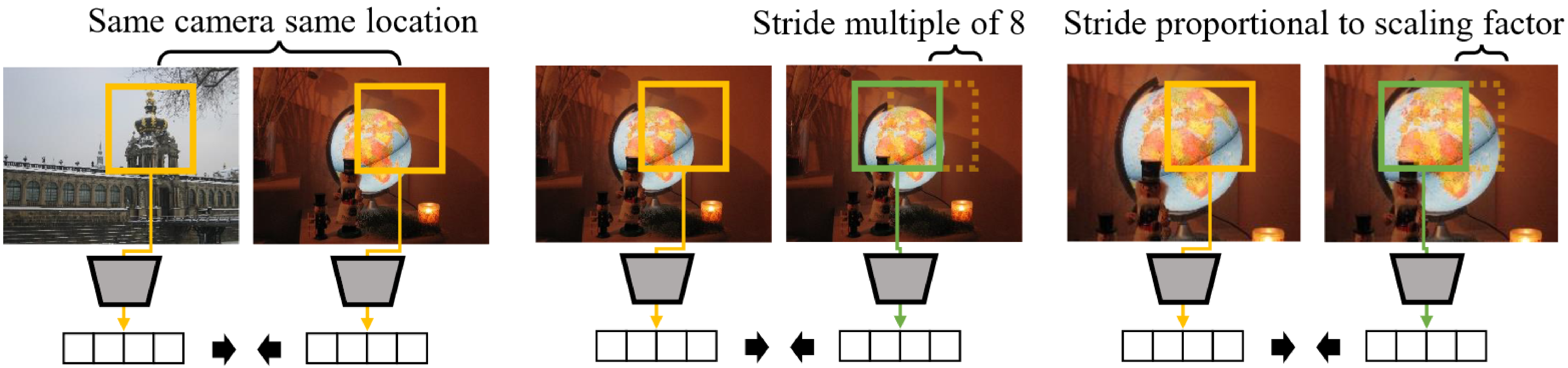


Our model leverages on statistical differences as well as spatial inconsistencies of these signatures in detecting forgeries

Contrastive Learning



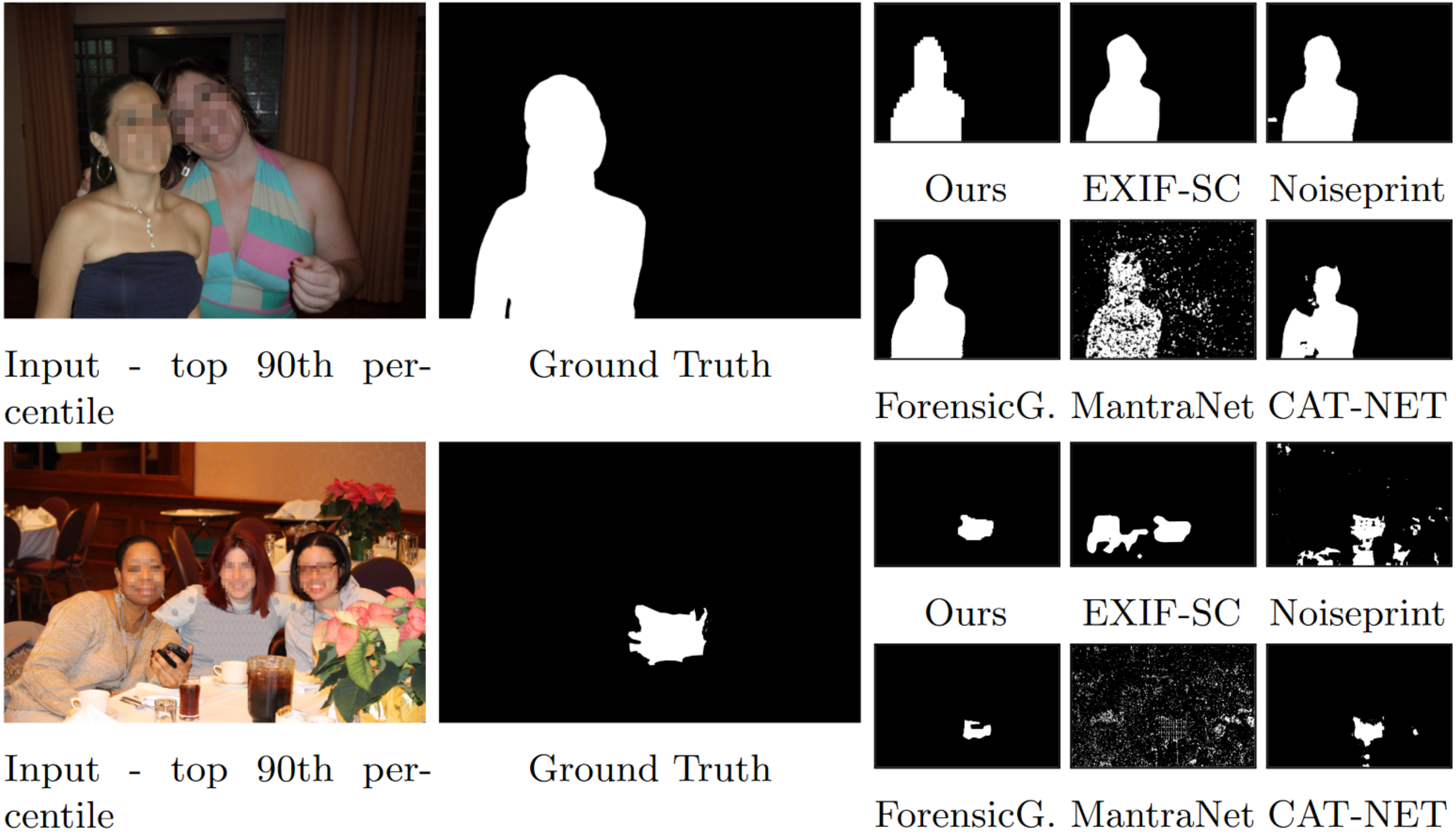
Contrastive Learning



(a) Learning camera signatures (b) Learning JPEG signatures (c) Learning rescaling signatures

Table 1: Comparison of forgery localization performance (MCC). Numbers in parenthesis use an image-specific threshold tuned on the ground truth labels; other numbers do not. HLED [3] and C-RCNN [50] trained on a subset of NC16, while CAT-NET [28] trained on IMD2020. We grayed those numbers out and excluded them in computing the average. We highlighted the best scores in bold and italicized the second best. For our method, the standard deviation is measured over 5 runs.

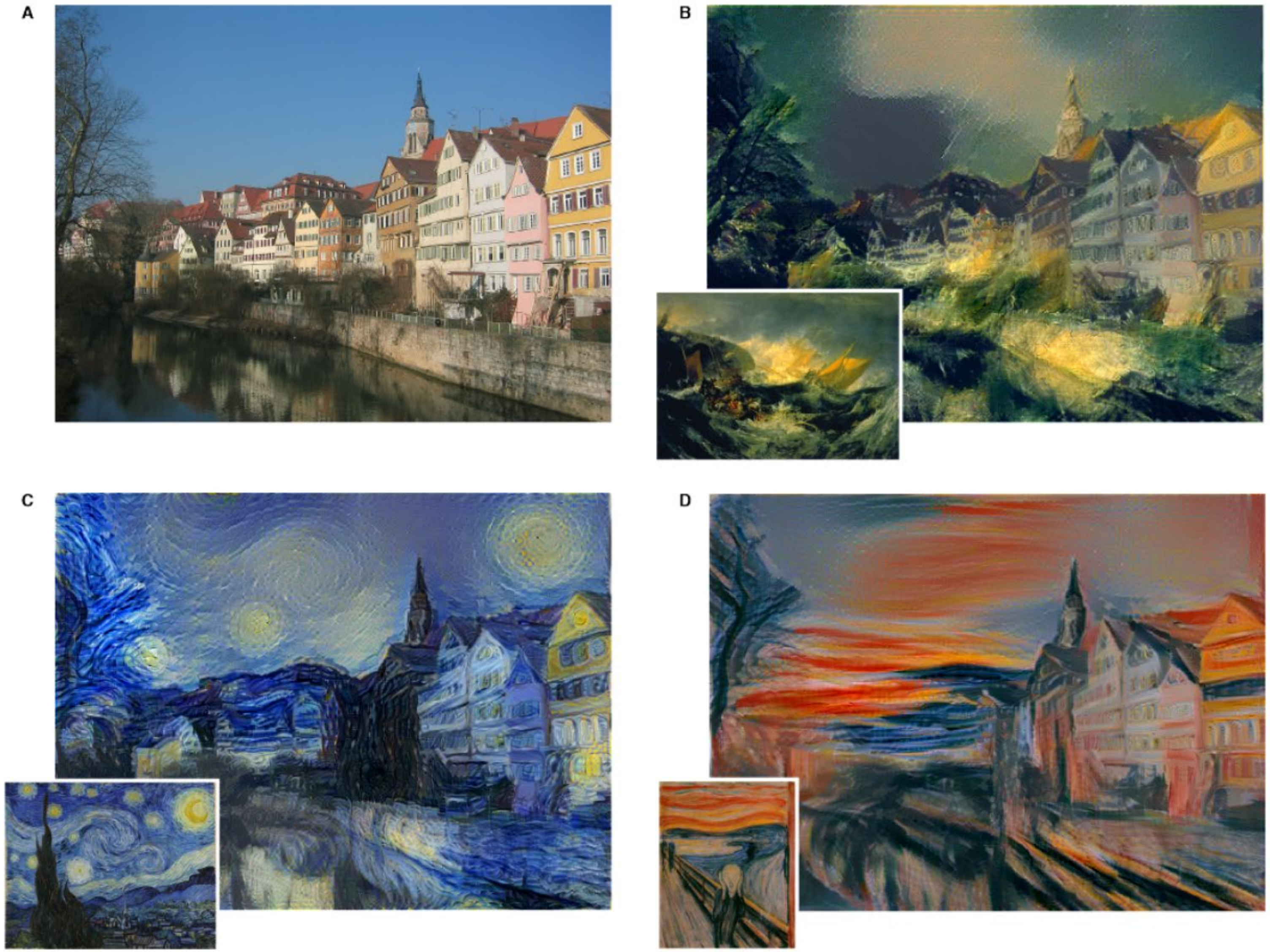
	Avg.	DSO-1	NC16	NC17-dev	RT	MFC18	IMD2020
ManTraNet [48]	19.8 (25.0)	41.8 (46.7)	11.6 (16.3)	14.8 (19.7)	19.1 (24.2)	10.2 (14.8)	21.6 (28.6)
GSRNet [53]	24.8 (34.1)	28.7 (46.2)	31.1 (40.9)	19.3 (22.7)	28.9 (36.8)	14.8 (20.8)	25.9 (37.1)
EXIF-SC [23]	24.9 (36.1)	41.0 (52.9)	25.7 (35.5)	**29.2 (41.7)**	17.0 (27.8)	18.2 (26.1)	18.4 (32.7)
InfoPrint [20]	-	55.0 (69.0)	28.0 (40.0)	25.0 (38.0)	-	-	-
Noiseprint [13]	31.8 (42.7)	70.1 (75.8)	28.1 (38.7)	24.6 (36.1)	21.8 (34.5)	23.9 (33.4)	22.2 (37.4)
ForensicGraph [36]	33.8 (41.1)	75.1 (80.2)	27.2 (35.2)	28.6 (36.9)	*31.0 (38.0)*	16.1 (23.2)	24.6 (33.4)
HLED [3]	20.8 (26.5)	18.2 (22.5)	40.4 (45.4)	14.1 (20.3)	16.7 (22.6)	14.3 (20.1)	21.4 (28.1)
C-RCNN [50]	18.4 (22.9)	21.2 (26.5)	93.1 (94.3)	23.8 (26.3)	14.9 (18.5)	14.4 (18.0)	17.7 (25.1)
CAT-Net [28]	*38.4 (45.4)*	*75.3 (80.5)*	**44.4 (56.5)**	21.6 (26.3)	20.4 (23.9)	**30.6 (39.9)**	88.8 (92.7)
CAT-Net (no qtable)	34.2 (39.4)	*75.3 (80.5)*	30.1 (36.9)	21.4 (25.6)	20.4 (23.9)	23.9 (30.4)	88.8 (92.7)
Ours	**39.4 (48.1)** ±0.92 (±0.68)	**85.7 (90.7)** ±1.51 (±0.73)	*35.4 (41.7)* ±1.25 (±0.97)	*28.9 (40.9)* ±0.76 (±0.84)	**34.7 (41.5)** ±0.72 (±0.54)	*24.3 (35.5)* ±0.69 (±0.56)	**27.7 (37.5)** ±0.58 (±0.43)


Input - top 90th percentile
Ground Truth
Ours
EXIF-SC
Noiseprint
ForensicG.
MantraNet
CAT-NET
Input - top 90th percentile
Ground Truth
Ours
EXIF-SC
Noiseprint
ForensicG.
MantraNet
CAT-NET

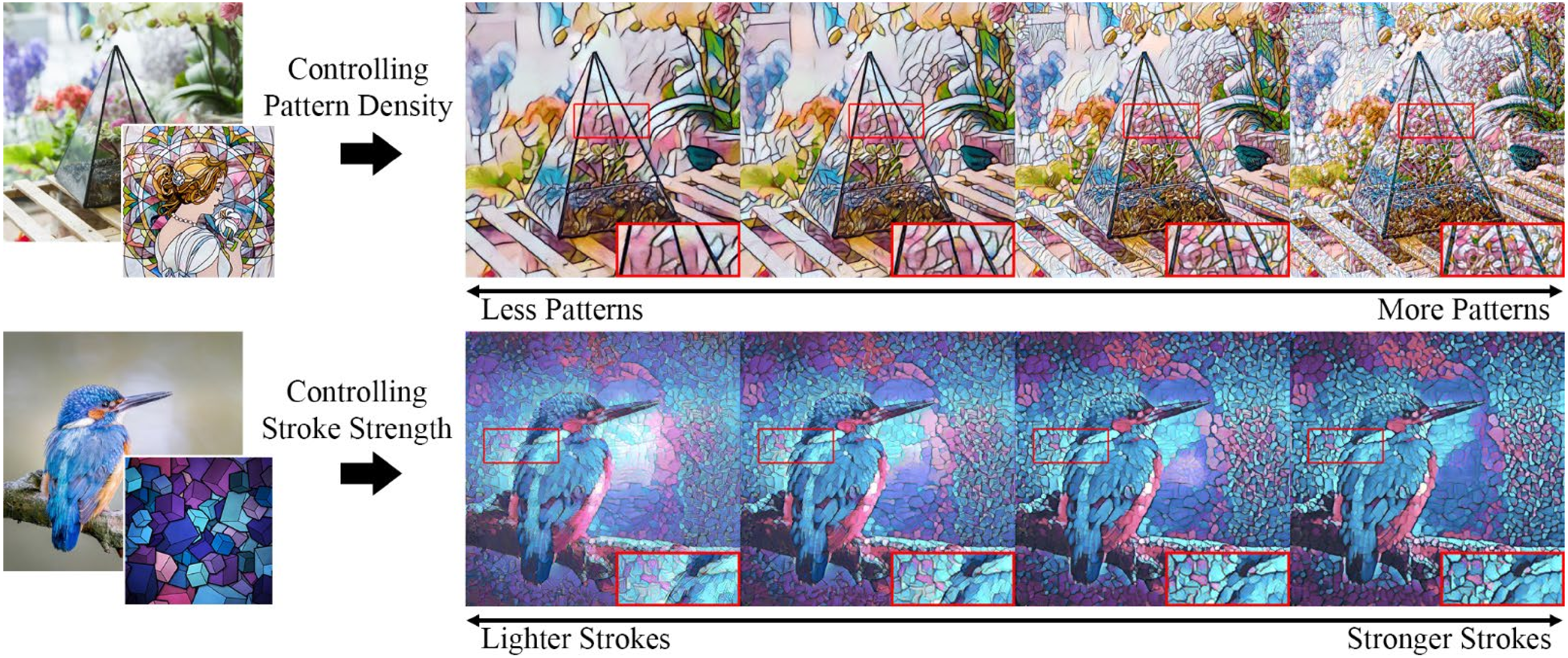
(Controllable) Style Transfer

Neural Style Transfer

Can apply new styles to other images BUT does not allow for any artistic control



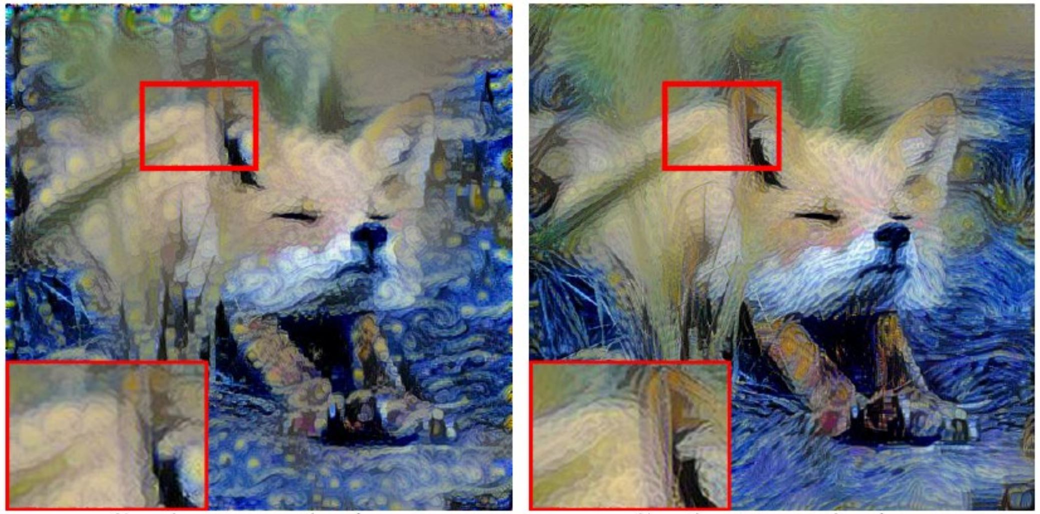
Density and Stroke Control



One way to control the size and density of patterns is to change the style resolution / receptive field

Surprisingly not as straightforward!

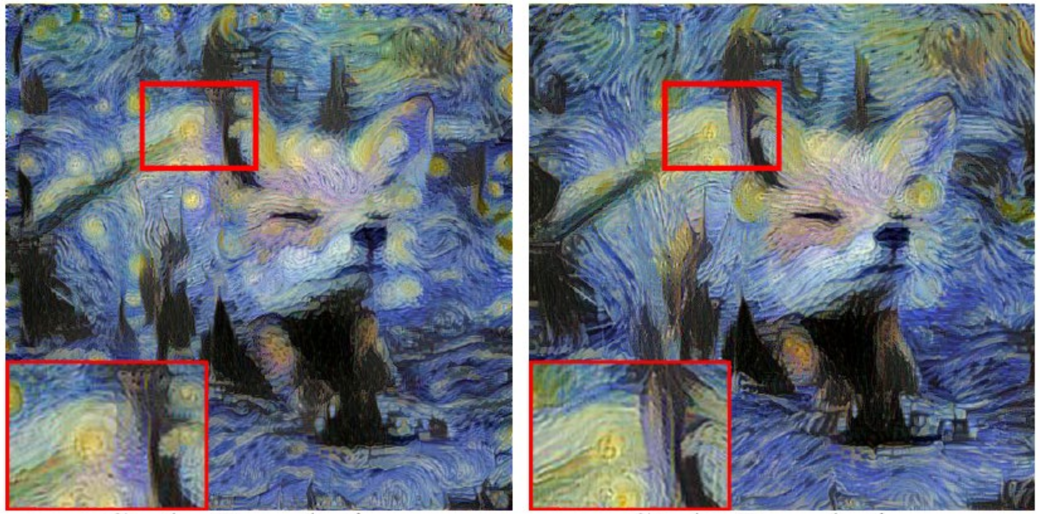
Ghosting effect!



Style Resolution 256×256 | Style Resolution 512×512

(a) Gram Matrix [5]

XX^T

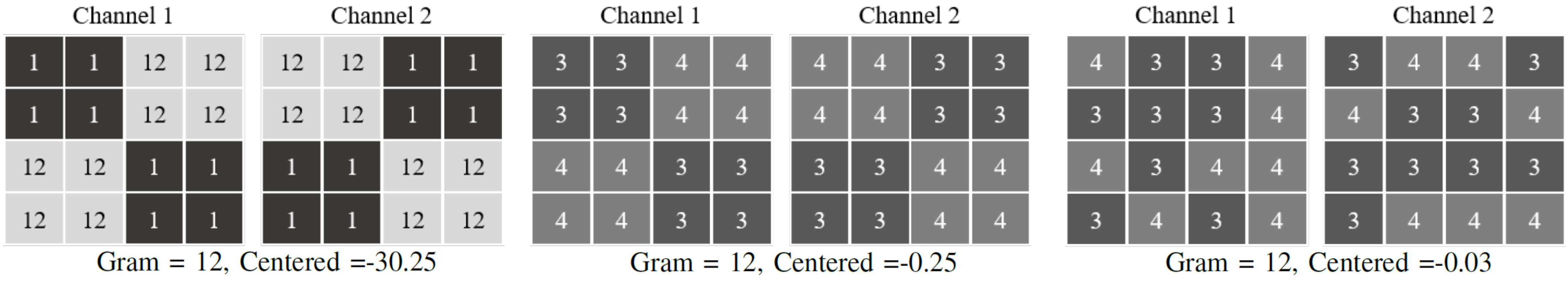


Style Resolution 256×256 | Style Resolution 512×512

(b) **Ours - Centered**

(covariance)

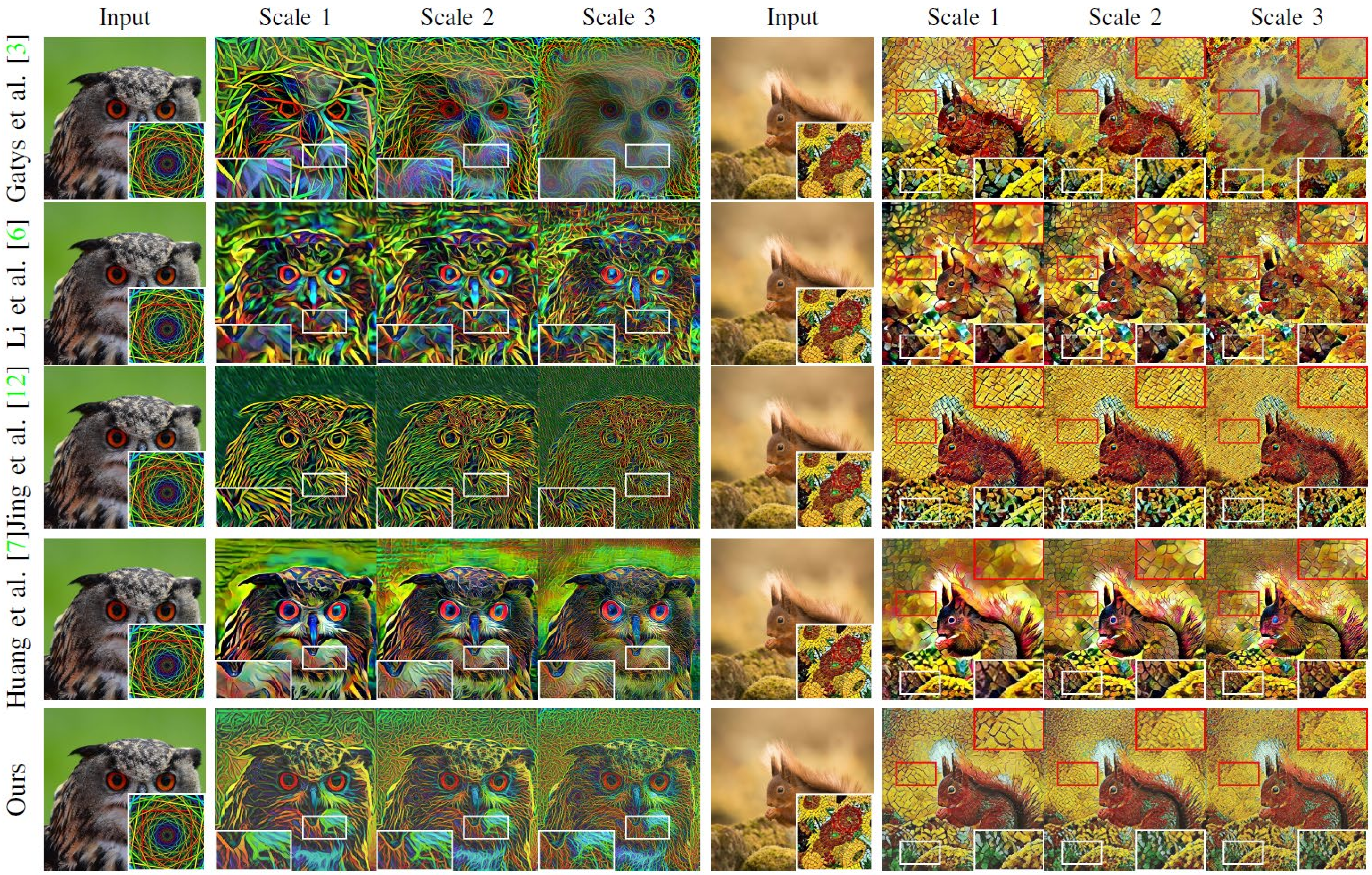
$(X - \mu_X)(X - \mu_X)^T$



(a) Strong Negative Correlation

(b) Weak Negative Correlation

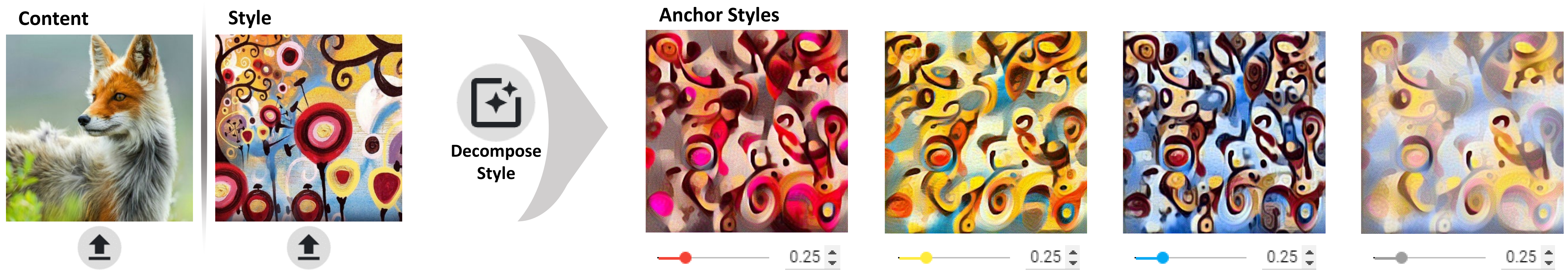
(c) No Correlation (Random Shuffle)

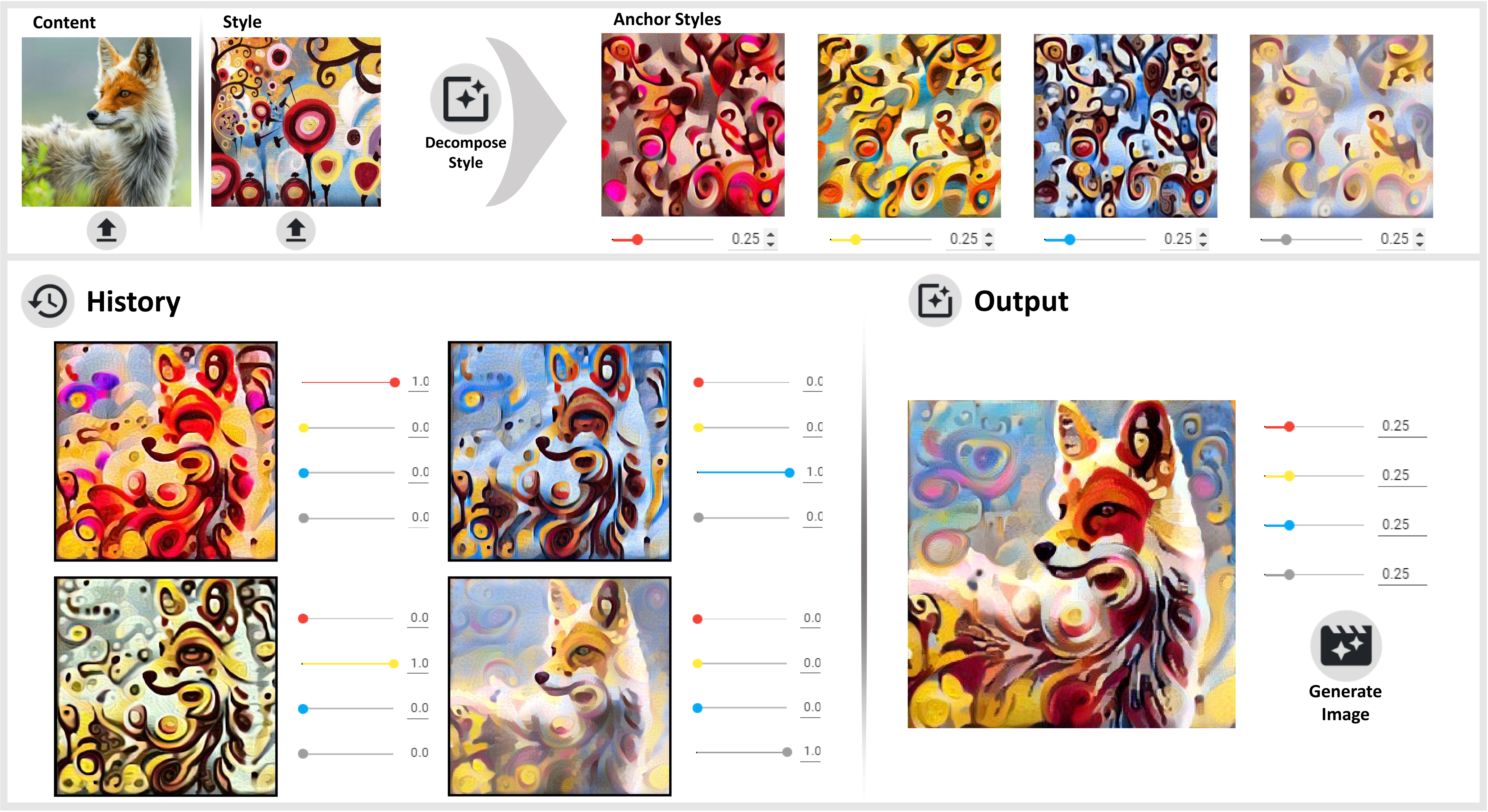


Input
Scale 1
Scale 2
Scale 3
Input
Scale 1
Scale 2
Scale 3
Gatys et al. [3]
Li et al. [6]
Jing et al. [12]
Huang et al. [7]
Ours

Neural Style Palette

Can we decompose a style image into "sub-styles"?




Content
Style
Decompose Style
Anchor Styles
0.25
0.25
0.25
0.25
History
1.0
0.0
0.0
0.0
0.0
0.0
1.0
0.0
0.0
1.0
0.0
0.0
0.0
0.0
0.0
1.0
Output
0.25
0.25
0.25
0.25
Generate Image

Work in Progress:
Detecting and counting Crop Pests

Goals:
- Crop pest and disease monitoring and surveillance (Early warning system)
- Assess efficacy of treatment plans (currently done with visual inspection)
- More precise treatment plans

Let me know if you want to collaborate!
Thank you!

Daniel Stanley Tan