삼성 DS^{2}과정 프로젝트

Image Restoration Challenge

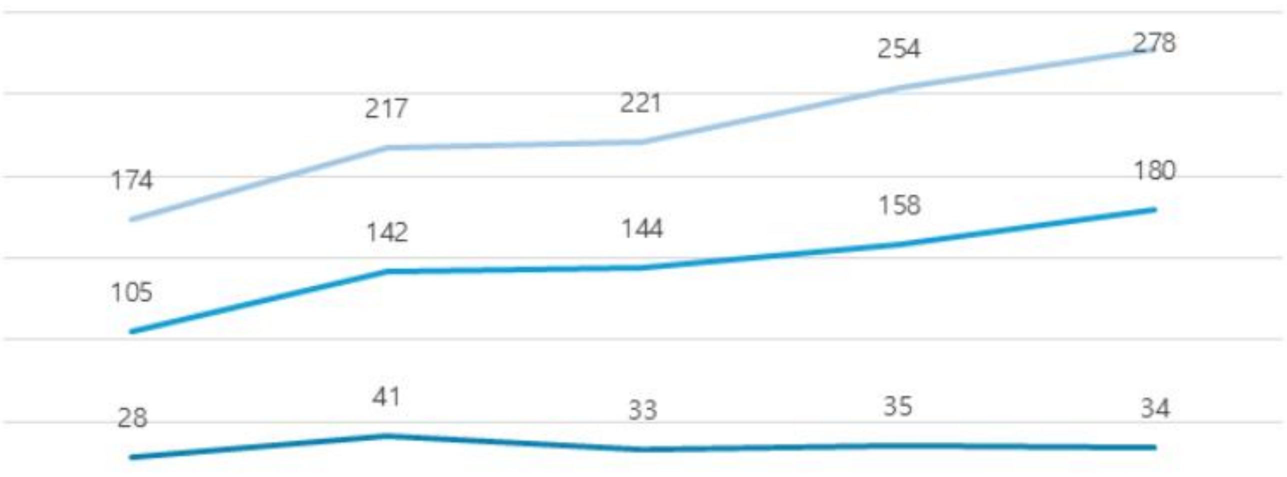
Jongho Lee, Ph.D

Laboratory for Imaging Science and Technology
Department of Electrical and Computer Engineering
Seoul National University

프로젝트, 프로젝트, 프로젝트 …

주관 서울대학교 공과대학 전기·정보공학부 | 후원 AIRS MEDICAL Klim ventures 서울대학교 뇌영상센터 | 협찬 서울대학교 공과대학 정보화

▪ SNU FastMRI Challenge



	2021	2022	2023	2024	2025
학과	28	41	33	35	34
팀	105	142	144	158	180
명	174	217	221	254	278

TOP 5

2021		2022		2023		2024		2025	
전기정보공학부	92	전기정보공학부	101	전기정보공학부	90	전기정보공학부	79	전기정보공학부	88
기계공학부	16	기계공학부	14	기계공학부	15	컴퓨터공학부	34	컴퓨터공학부	47
컴퓨터공학부	11	컴퓨터공학부	14	자유전공학부	14	기계공학부	31	첨단융합학부	31
자유전공학부	5	자유전공학부	10	컴퓨터공학부	12	첨단융합학부	13	기계공학부	26
물리천문학부	4	물리천문학부	8	조선해양공학과	10	자유전공학부	12	자유전공학부	15
화학교육과	4								
원자핵공학과	4								

2025년

참여학과: **34개 학과, 180팀**
참여인원: **278명**

전기정보공학부	88
컴퓨터공학부	47
첨단융합학부	31
기계공학부	26
자유전공학부	15
재료공학부	7
원자핵공학과	6
의학과	6
건설환경공학부	5
수리과학부	5
통계학과	5
항공우주공학과	5
기계항공공학부	3
건축학과	2
물리교육과	2
생명과학부	2
언어학과	2
에너지자원공학과	2
조선해양공학과	2
체육교육과	2
화학생물공학부	2
간호학과	1
경제학부	1
농경제사회학부	1
바이오시스템소재학부	1
산업공학과	1
생물교육과	1
소비자아동학부	1
수의학과	1
식품동물생명공학부	1
식품영양학과	1
의류학과	1
지구환경과학부	1
화학부	1

~~제5회~~ 제1회

2025 ~~SNU FastMRI Challenge~~ 삼성DS²과정 Digital Image Processing Challenge

~~FastMRI Challenge~~

Semiconductor Image Processing Challenge

참가대상 ~~서울대 학부생(1팀 최대 2명)~~ 삼성DS²과정생

대회기간 ~~2025.07.01(화) - 2025.08.20(수)~~

참가방법 온라인접수 (fastmri.snu.ac.kr)

문의방법 이메일문의 (fastmri.snu@gmail.com)

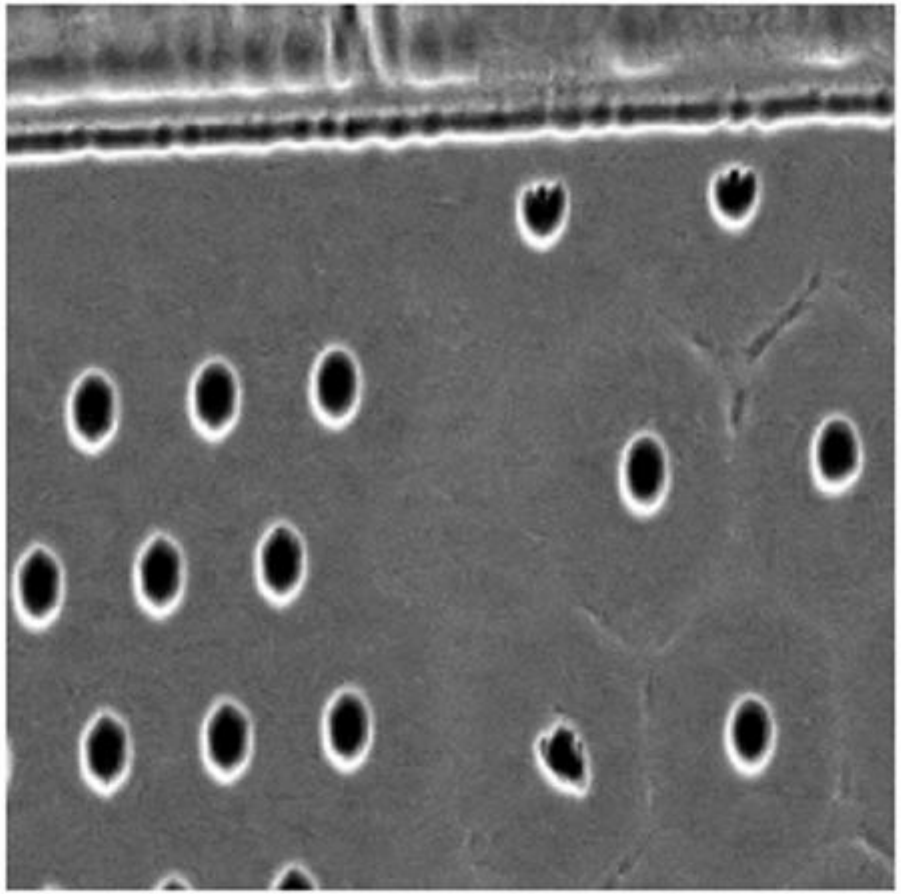
상 금 ~~총 2,000만원(1등 1,000만원)~~ To be announced

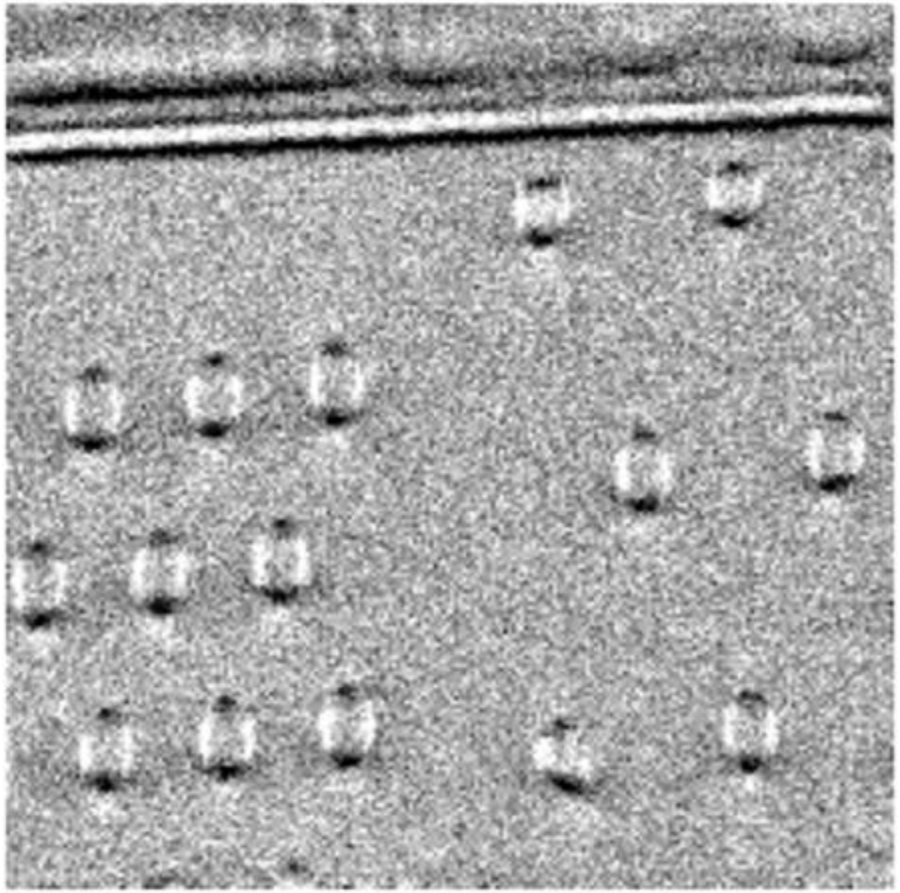
괜찮아요! Deep Learning 몰라요? MRI 몰라요?

→ Tutorial + Q&A 있습니다~

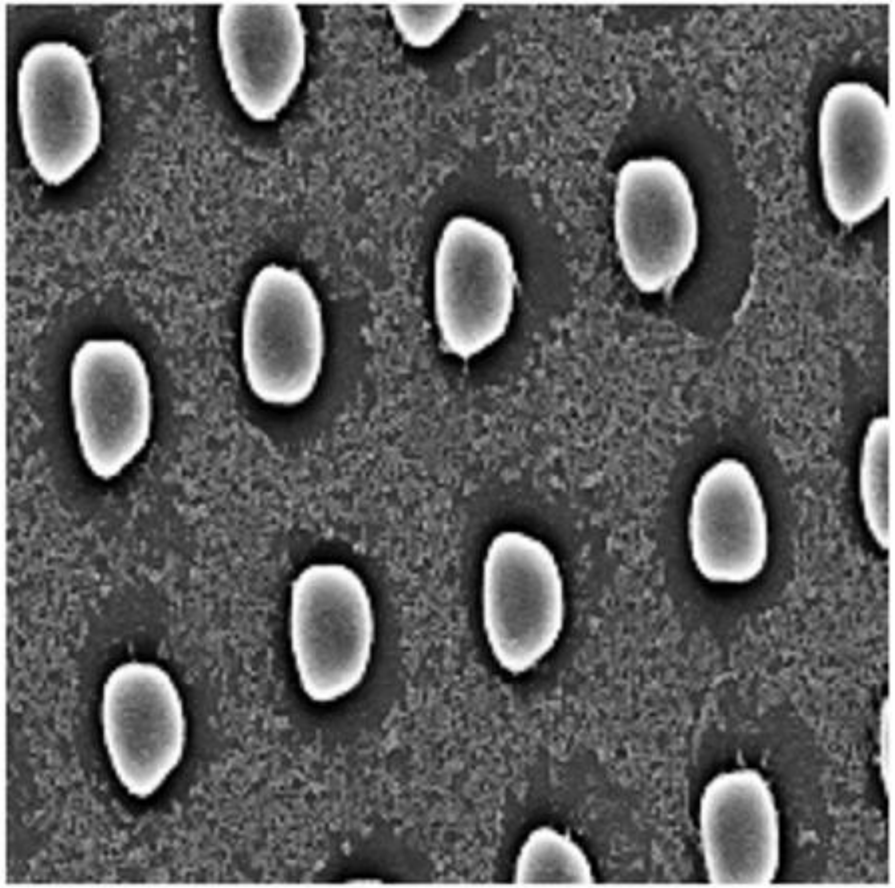
주관 서울대학교 공과대학 전기·정보공학부 | 후원 AIRS MEDICAL Klim ventures 서울대학교 뇌영상센터 | 협찬 서울대학교 공과대학 정보화

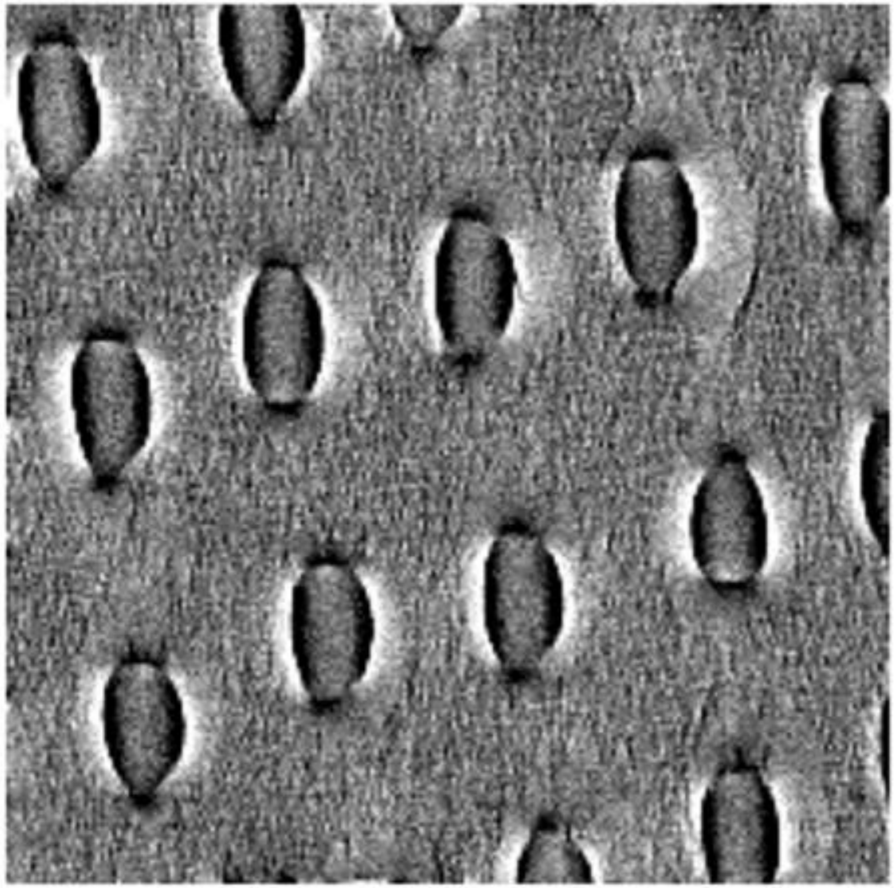
Aim of our challenge



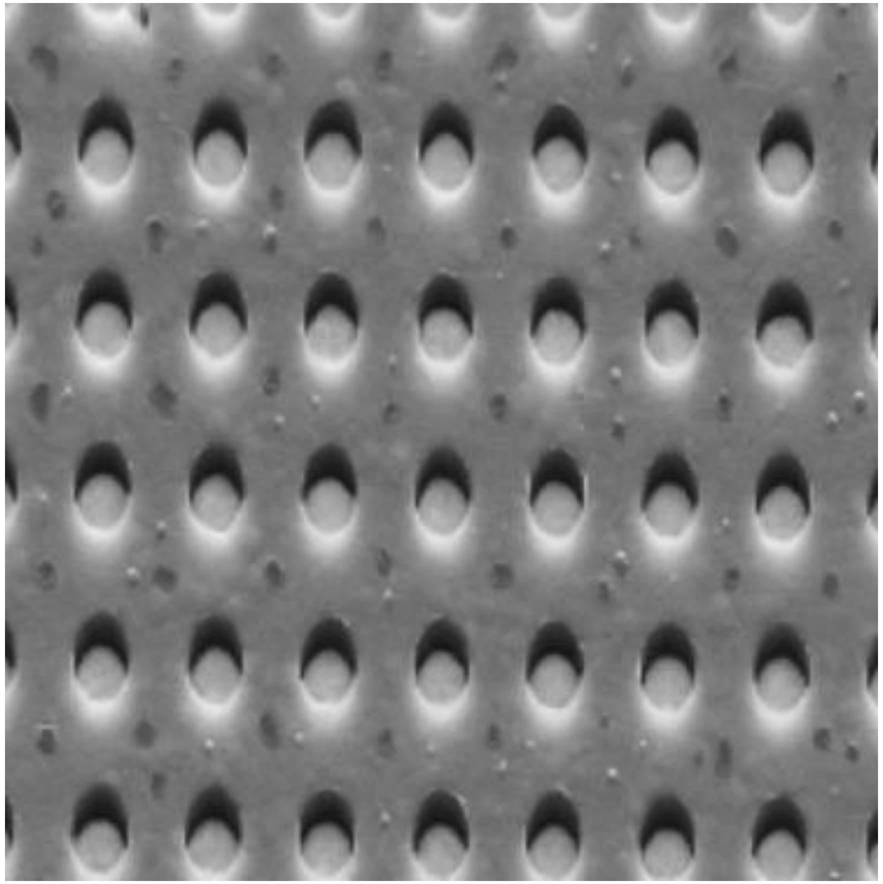


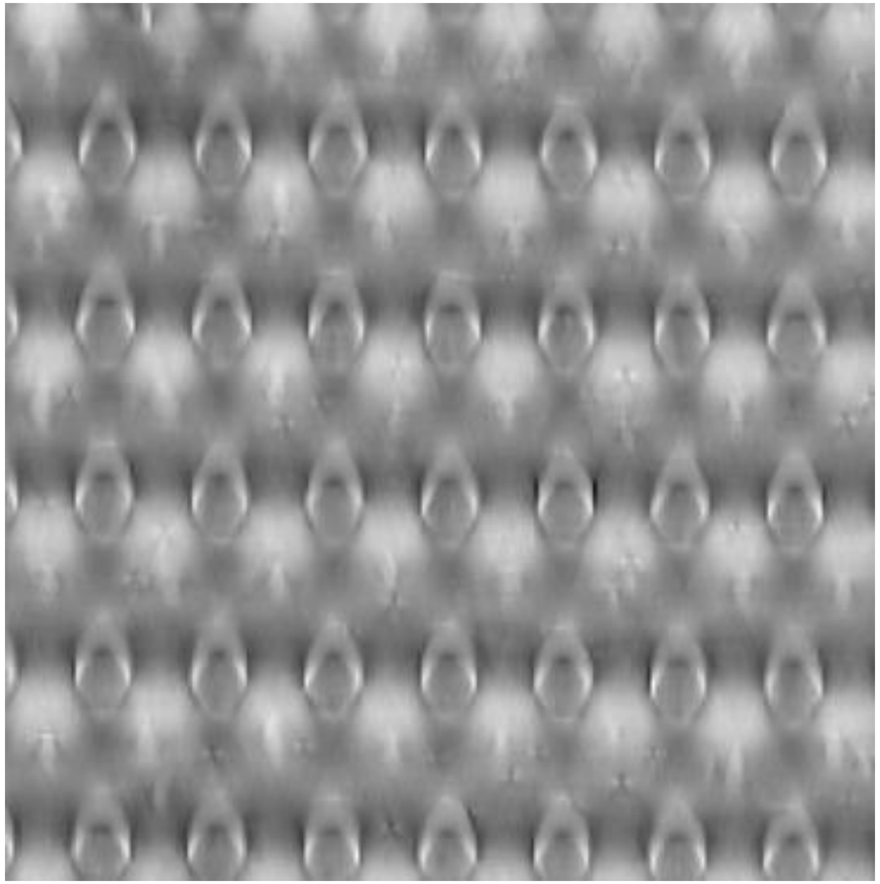
Aim of our challenge



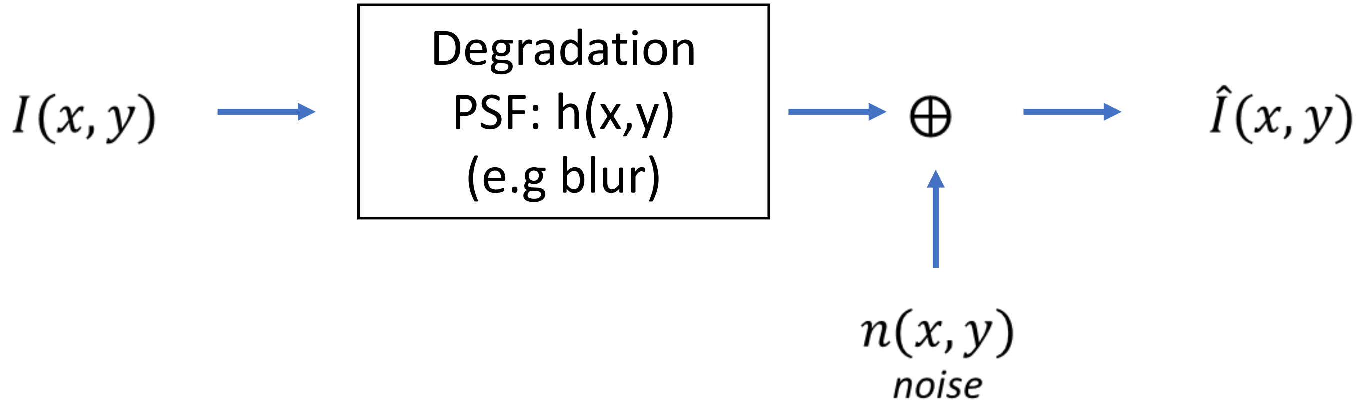


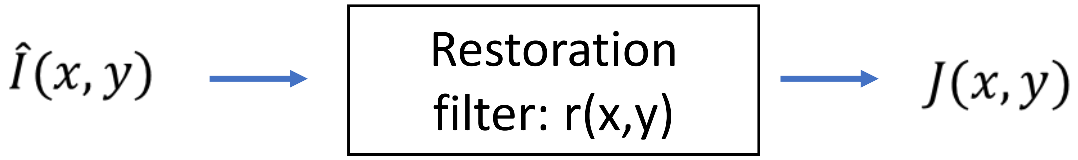
Aim of our challenge





Basic model for degradation





$$\hat{I}(x,y) = I(x,y) * h(x,y) + n(x,y) \quad J(x,y) = \hat{I}(x,y) * r(x,y) + n(x,y) * r(x,y)$$

$$\hat{\boldsymbol{I}}(u,v) = \boldsymbol{I}(u,v)\boldsymbol{H}(u,v) + \boldsymbol{N}(u,v) \quad \boldsymbol{J}(u,v) = \hat{\boldsymbol{I}}(u,v)\boldsymbol{R}(u,v) + \boldsymbol{N}(u,v)\boldsymbol{R}(u,v)$$

Easier case

- H = identity (no blur)

  - Degraded image only contains additive noise.

- Noise is typically described by a PMF

  $P(n(x, y) = z)$ assume I.I.D.

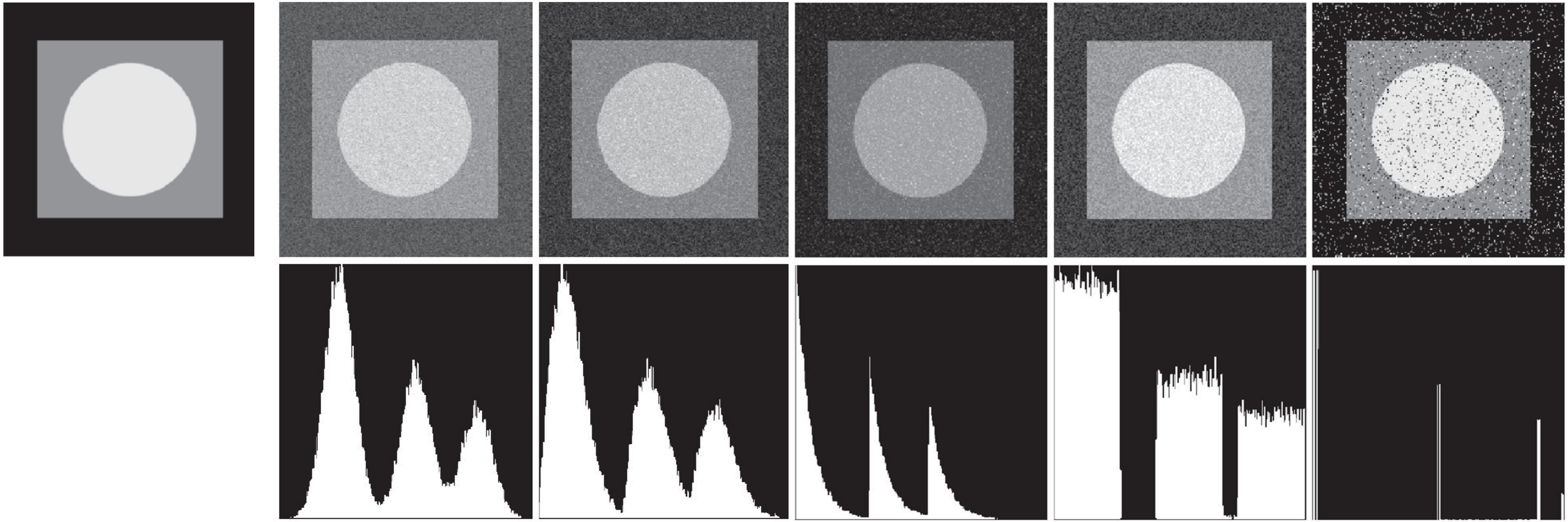
- Noise may be due to

  - Non-ideal sensor elements

  - Environmental conditions (light level, temp)

  - Corruption during transmission / compression

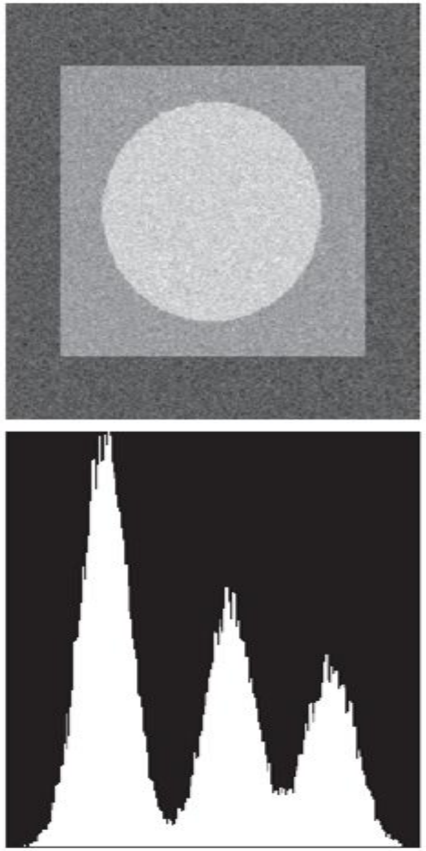
Noisy images



Noise distribution | Gaussian (Thermal noise) | Rayleigh | Exponential (laser imaging) | Uniform | Salt & pepper

Determine noise type and power?

- Typical ways
  - Find a region that should be flat (constant intensity)
  - Look at image histogram



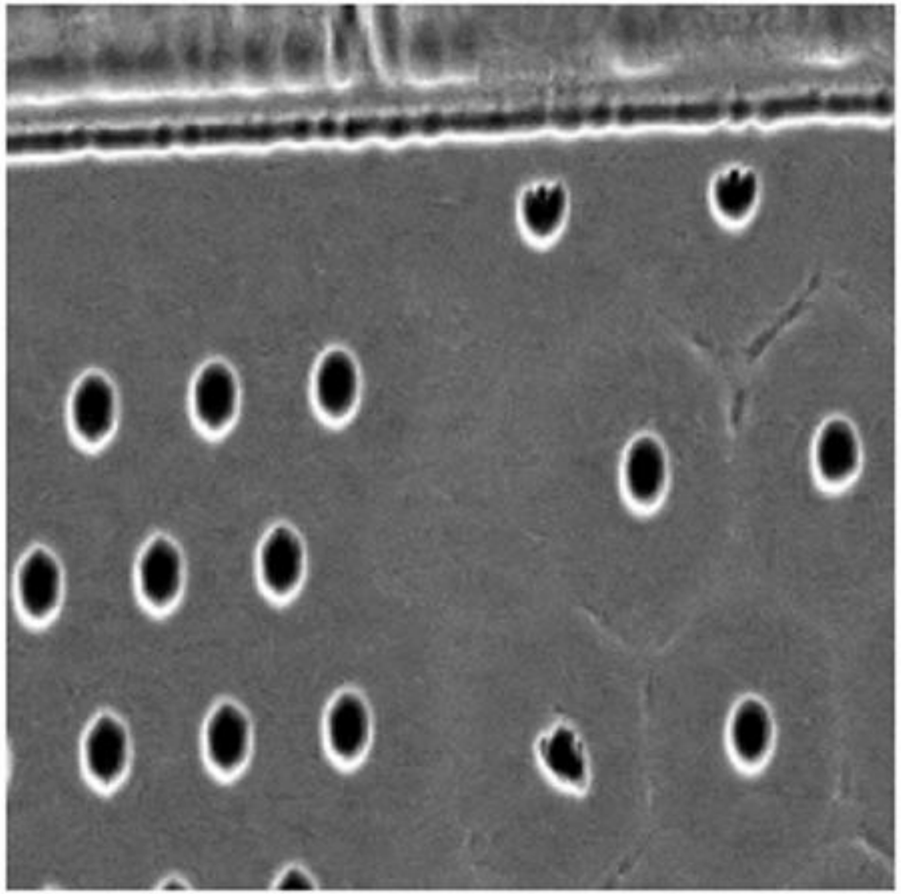
$$\hat{\mu} = \sum_{z_i \leftarrow S} z_i p_i(z_i)$$

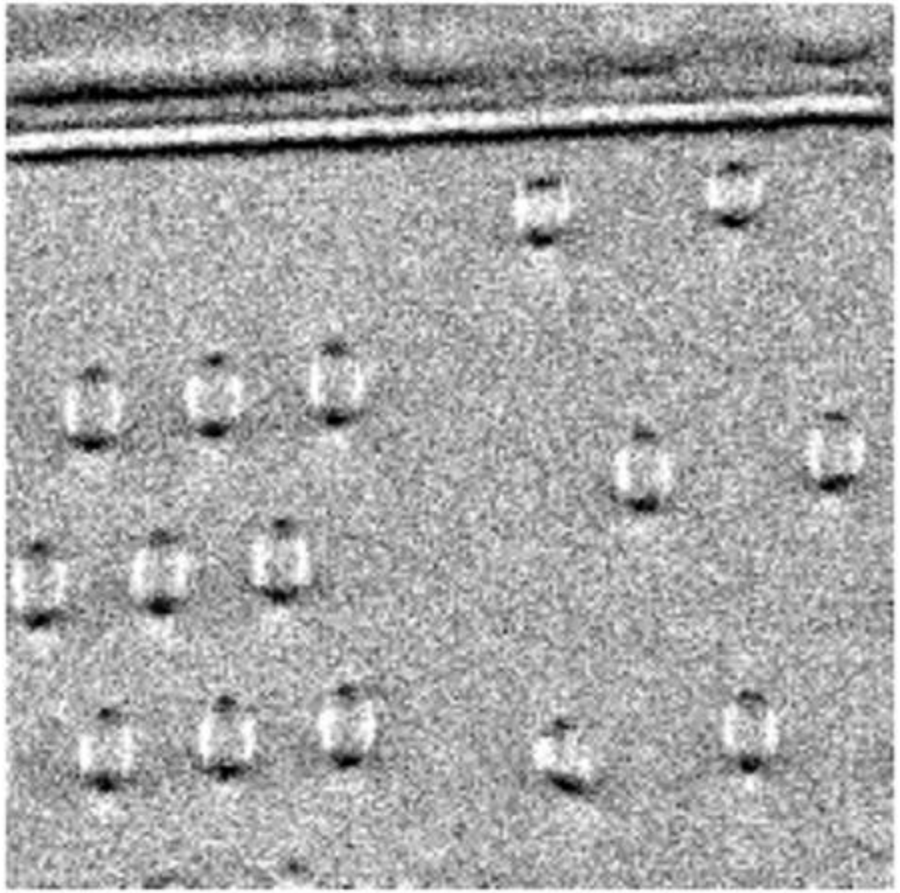
$$\widehat{\sigma^2} = \sum_{z_i \leftarrow S} (z_i - \mu)^2 p_i(z_i)$$

Denoising

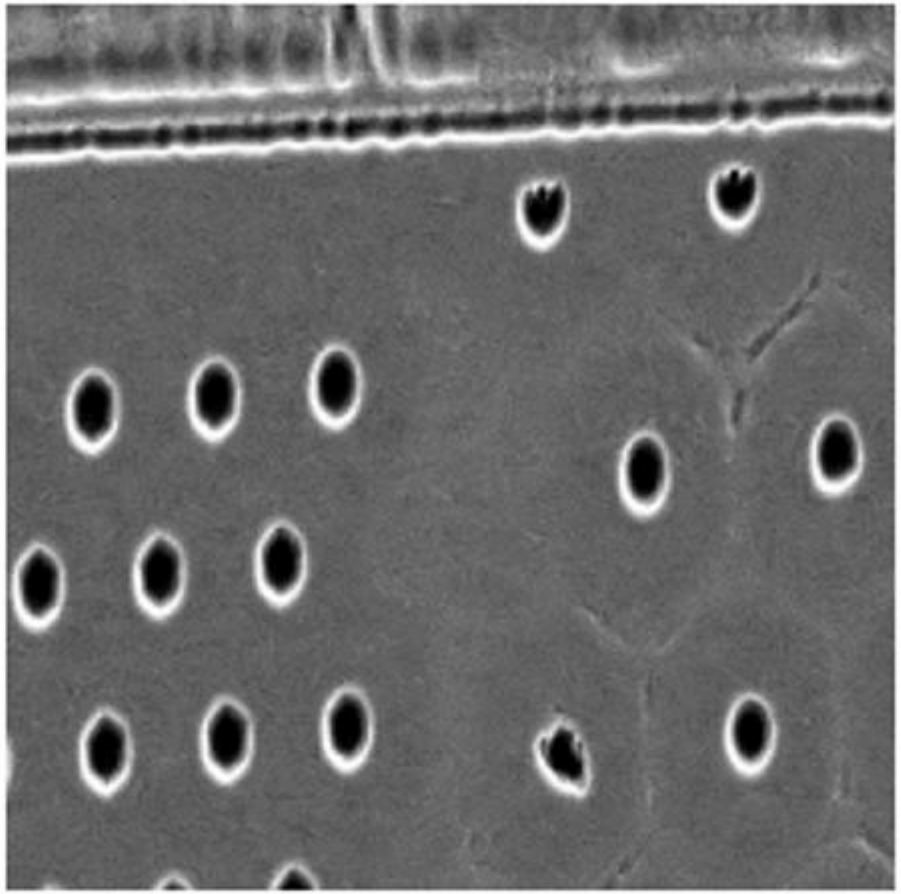
- Low pass filter (image domain, Fourier domain)
- Median filter for salt and pepper noise
- Adaptive filter
- Deep learning based denoising

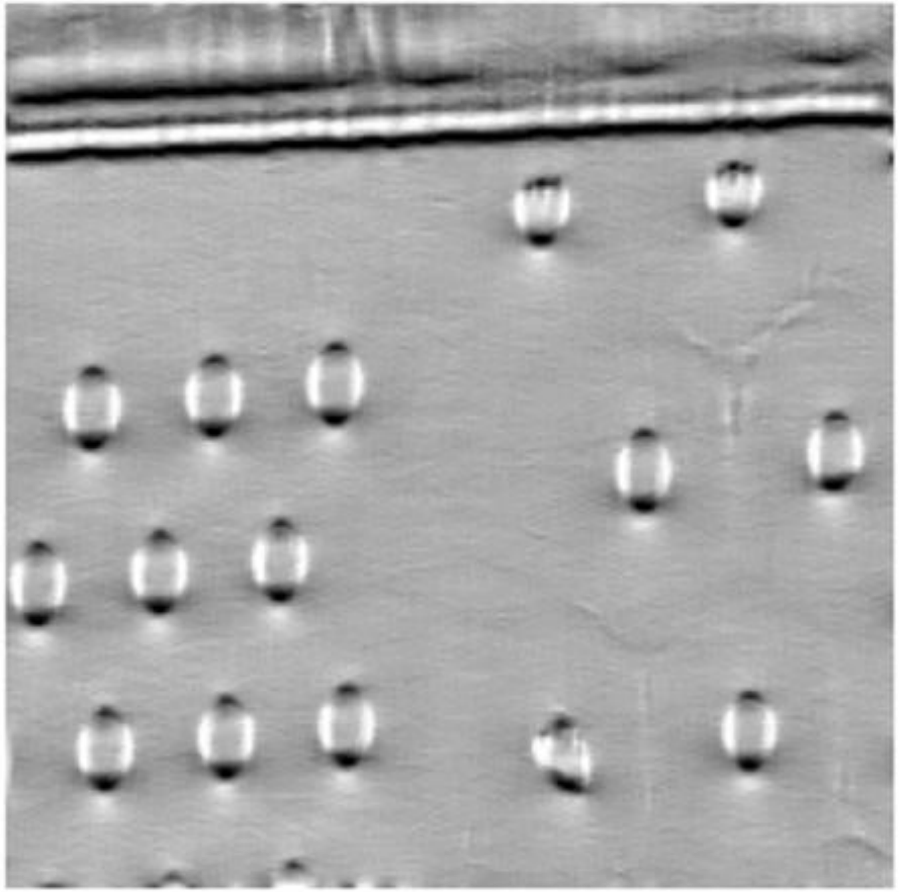
Aim of our challenge





Aim of our challenge





What happens when we also have degradation?

- $\hat{I}(x,y) \;=\; I(x,y) * h(x,y) \;+ n(x,y)$
- $\hat{\boldsymbol{I}}(u,v) \;=\; \boldsymbol{I}(u,v)\, \underbrace{\boldsymbol{H}(u,v)}_{\text{blur}} + \; \boldsymbol{N}(u,v)$

- How do we estimate $h(x,y)$?

What happens when we also have degradation?

- Guess: Take a piece of the degraded image and guess what the original image should have looked like

$$\hat{I}_s(u,v) \quad vs. \quad \hat{G}_s(u,v)$$

S = guess area — Manual guess

$$H(u,v) = \frac{\hat{I}_s(u,v)}{\hat{G}_s(u,v)}$$

- Experiment if you have access to the imaging device: Directly acquire the impulse response / point spread function

- Estimate $h(x,y)$ (e.g. Gaussian blur)

Inverse filtering

- We have the degraded image I(x,y)
- We have the estimated blur $h(x,y)$

- Inverse filtering

$$J(u,v) = \frac{\hat{I}(u,v)}{H(u,v)}$$

- Usually is bad. Why?

$$J(u,v) = \frac{I(u,v)H(u,v) + N(u,v)}{H(u,v)} = I(u,v) + \frac{N(u,v)}{H(u,v)}$$

※ If H(u,v) is very small for some (u,v) then $\frac{N(u,v)}{H(u,v)}$ is very large

⇒ poor reconstructions.

Wiener filter

- Wiener filter: minimum mean-square error filtering

$$e^2 = E\left\{\left(I(x,y) - \hat{I}(x,y)\right)^2\right\} \qquad S_F = |I(u,v)|^2 \; , S_n = |N(u,v)|^2$$

$$J(u,v) = \left[\frac{H^*(u,v)\, S_F(u.v)}{S_F(u,v)|H(u,v)|^2 + S_n(u,v)}\right]\hat{I}(u,v)$$

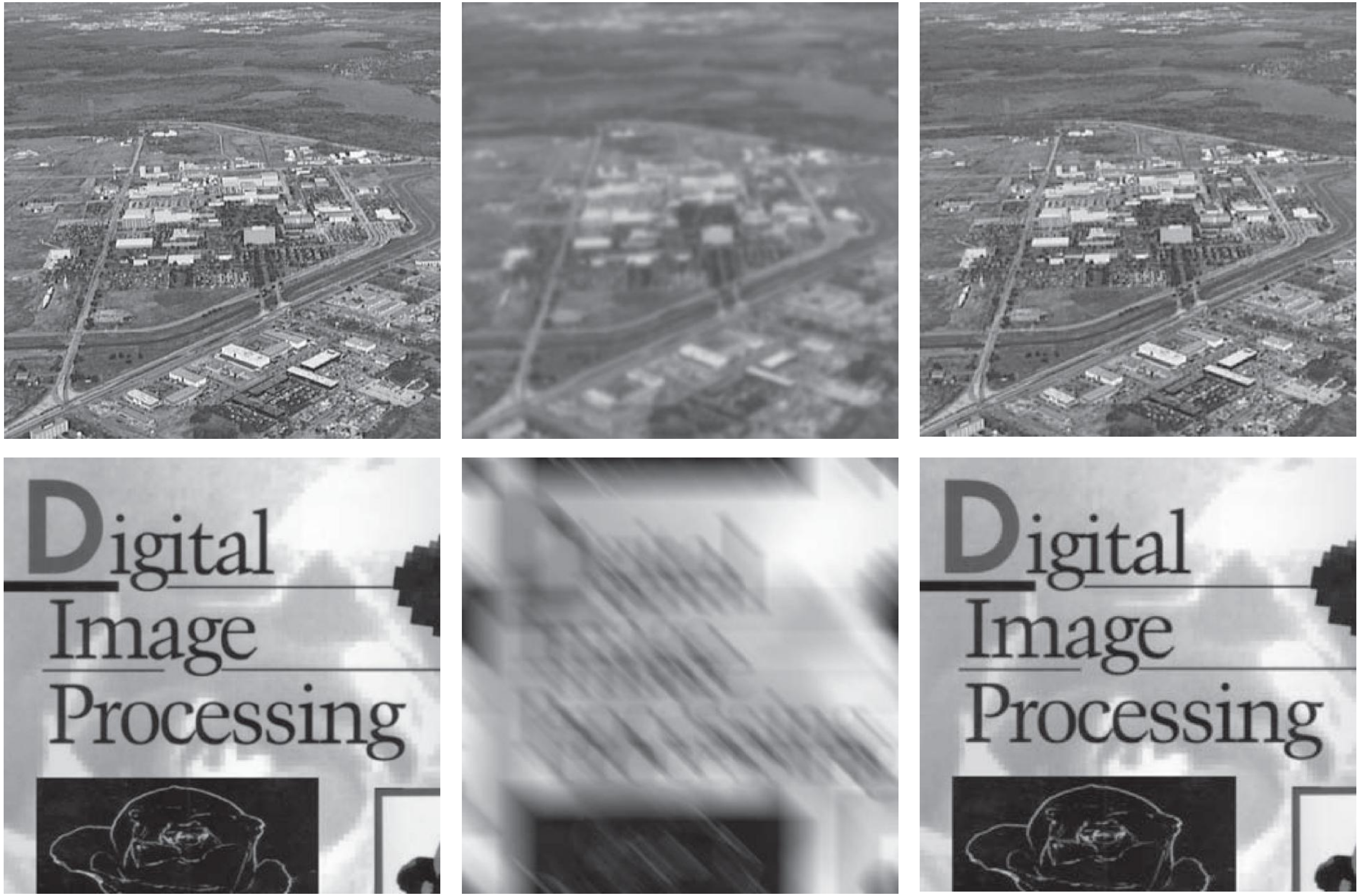
$$= \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + \frac{S_n(u,v)}{S_F(u,v)}}\right]\hat{I}(u,v)$$

- If we don't know $S_F(u,v)$ (requiring the original image), we use:

$$\bar{\bar{I}}(u.v) = \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + K}\right]\hat{I}(u,v)$$

Tunable parameter

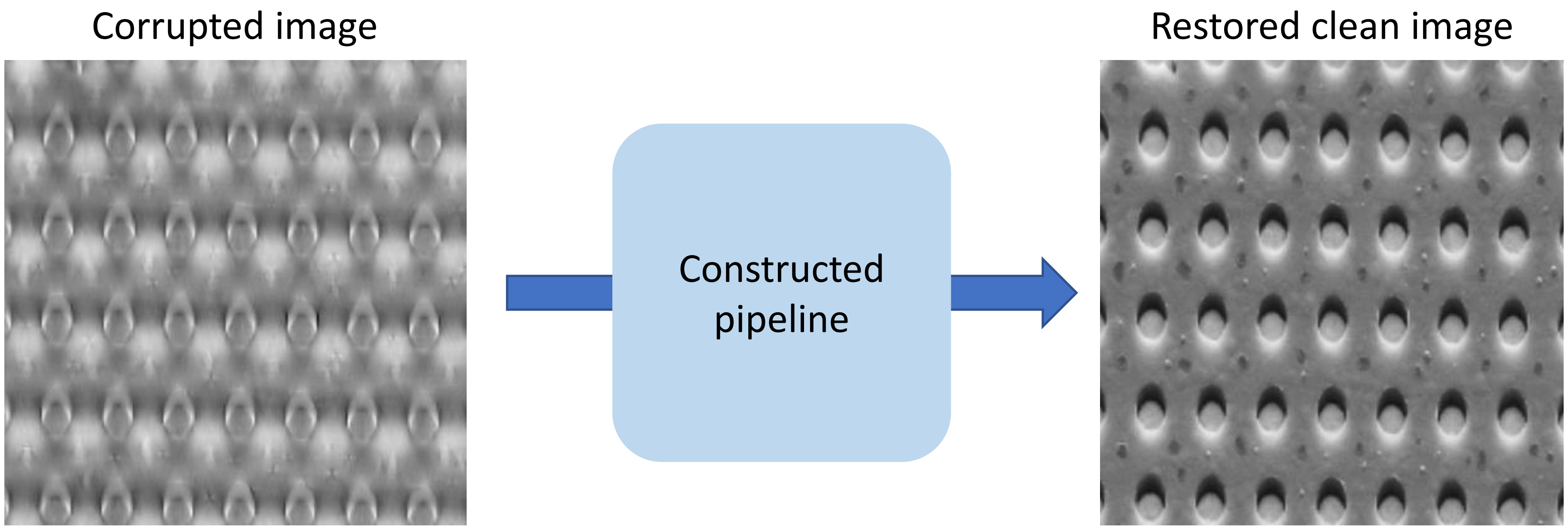
Wiener filter



Wiener filter

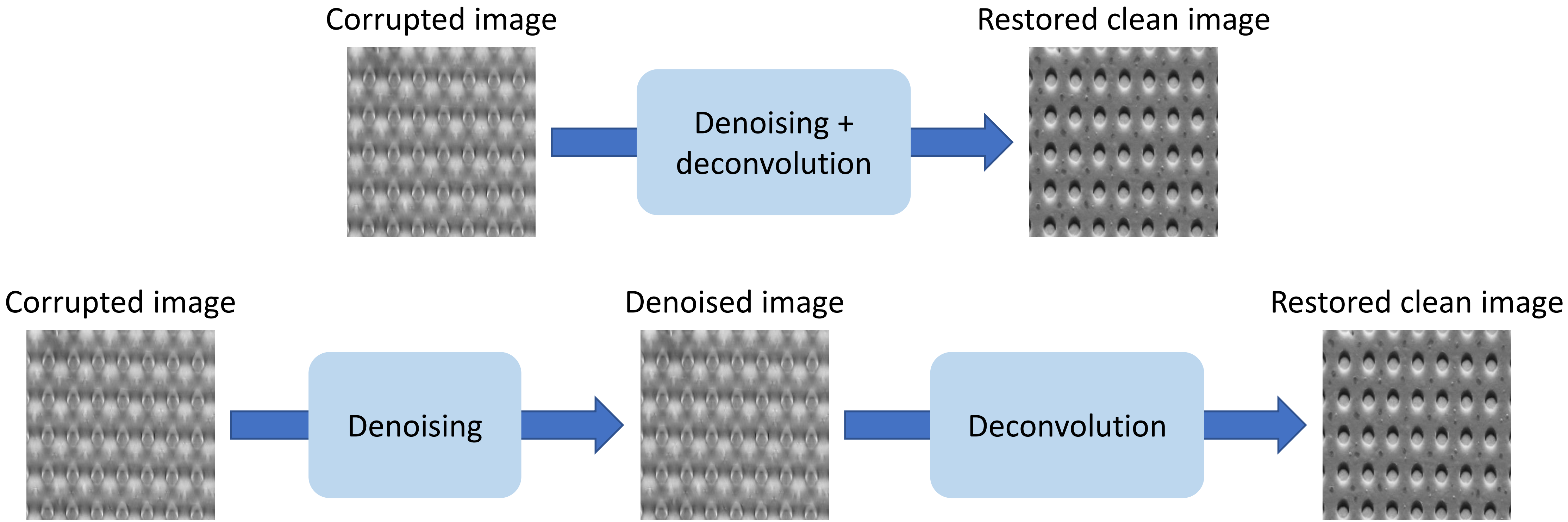
Project overview

- Goal: construct a pipeline to restore corrupted images into clean images
- Dataset
  - Clean gray-scale images
  - Corrupted images generated by
    - Convolution with a 2D kernel
    - Addition of noise



Project overview

- Potential restoration strategies:
  - Classical Wiener filter
  - End-to-end neural network: train a single network to restore clean
  - Two-stage neural network: train a denoising network and a deconvolution network separately

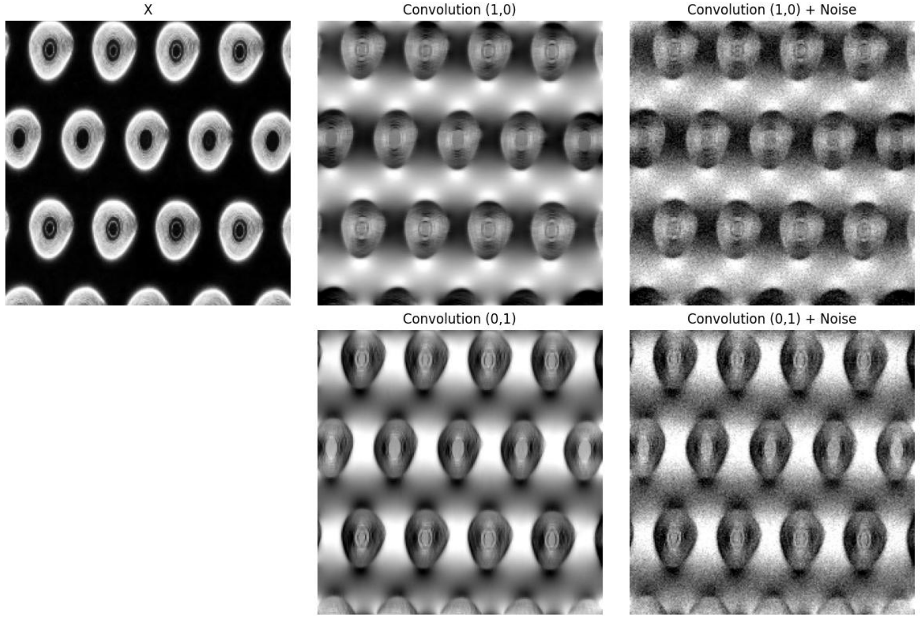


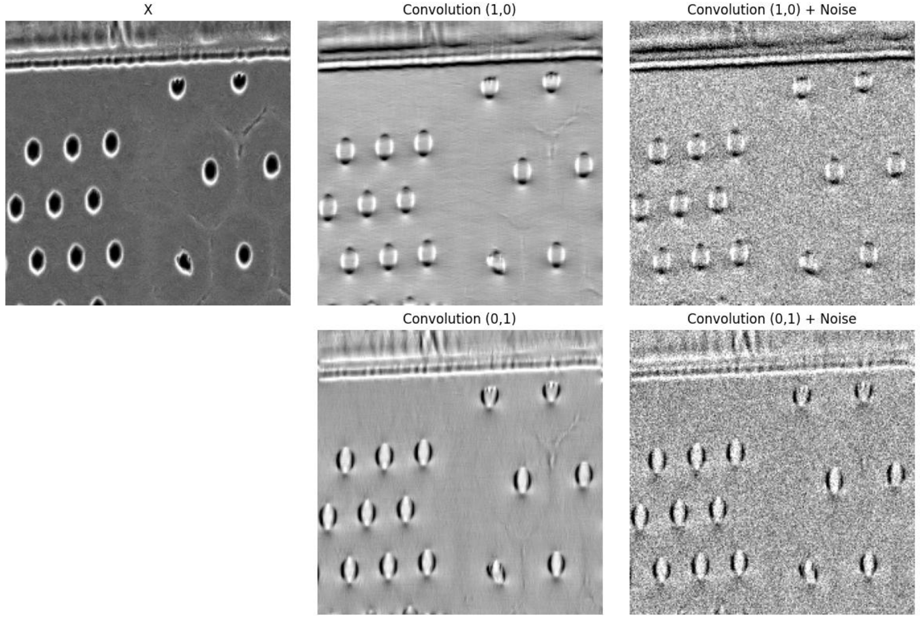
Challenge & approaches

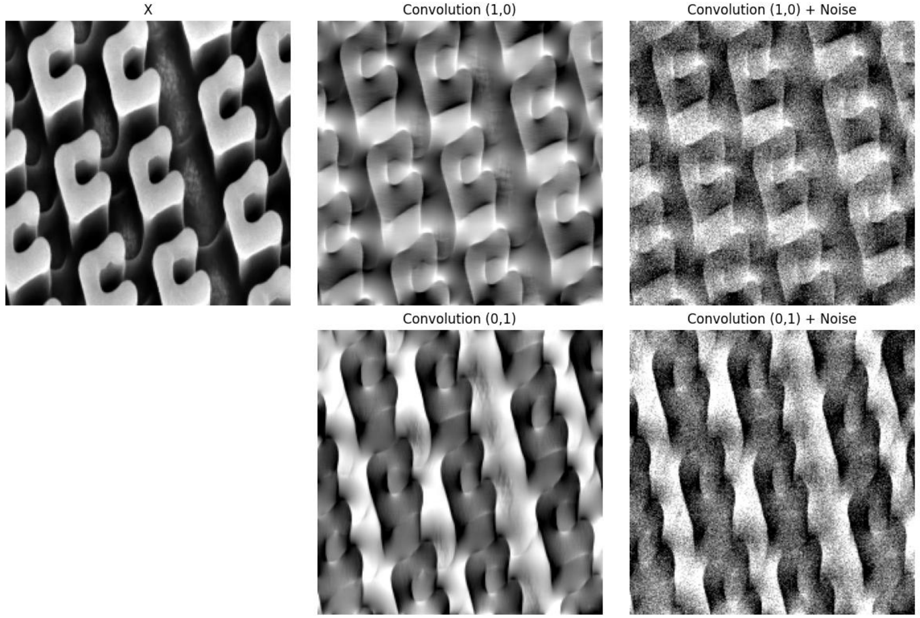
- In practical scenarios, clean ground-truth images are rarely available
  → A purely supervised approach is unrealistic in many applications

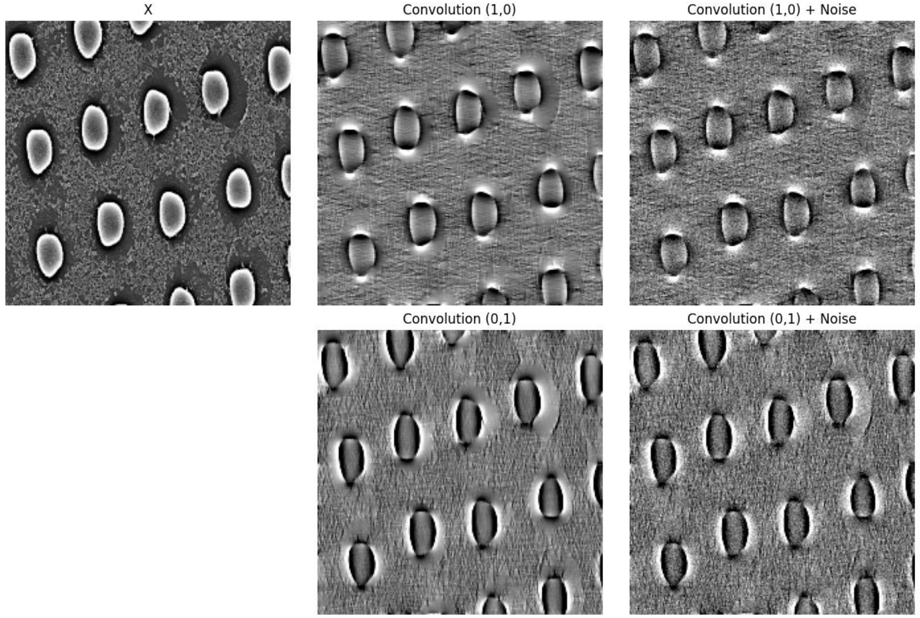
- Possible approaches to consider:
  - Denoising: self-supervised learning, filtering, …
  - Deconvolution: leveraging multi-orientation information, regularization, thresholding, …
  - General approach: generating pseudo labels, and any other creative ways

Multiple directional convolution images available for kernel estimation









Dataset

- Train / validation dataset:
  - Clean images: 7,368
  - Forward corruption model:
    - 2D dipole convolution (model is given)
    - Noise model (noise characteristic is not given)
  - For each clean image, you may synthesize corrupted variants, using the provided model

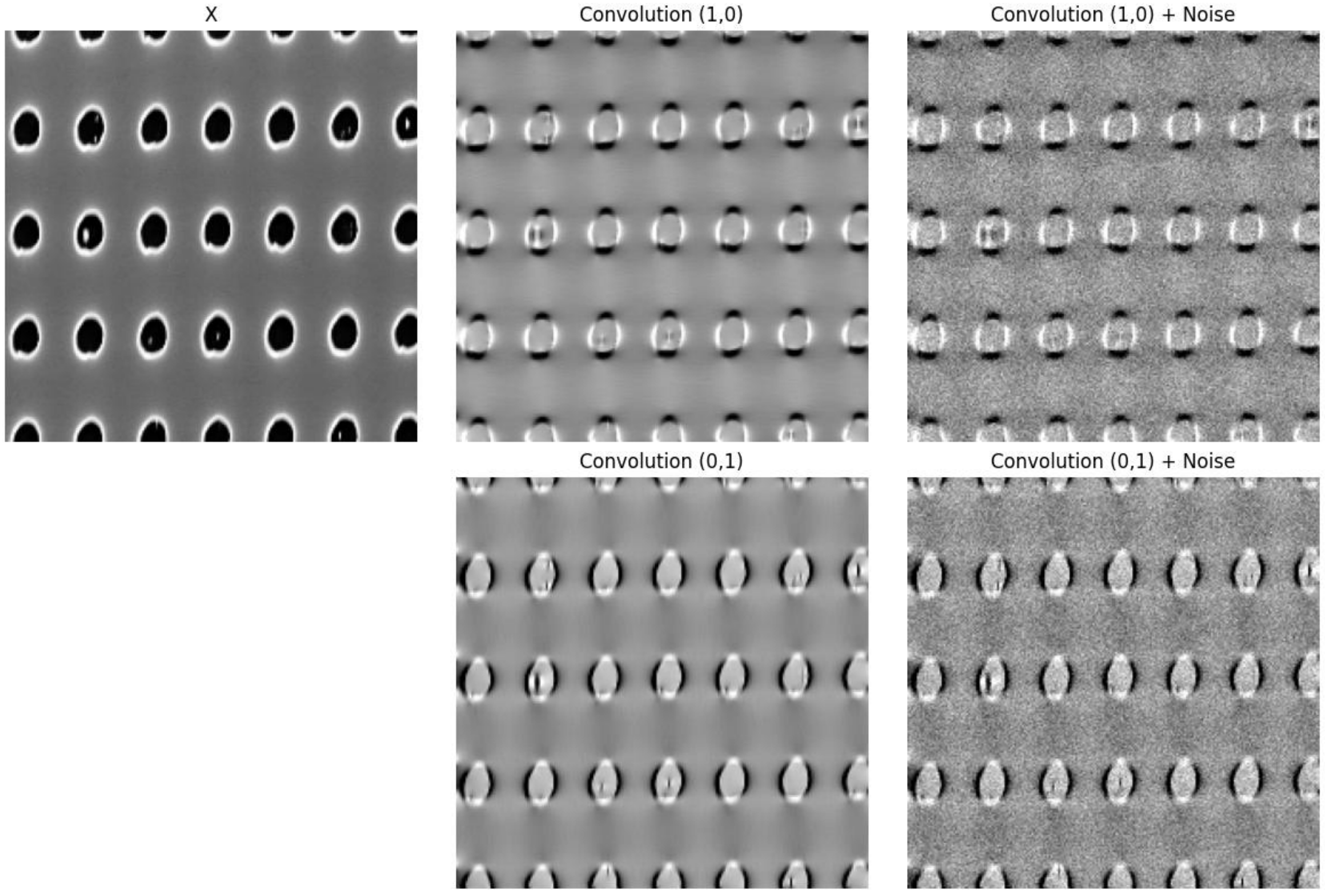
- Test dataset:
  - Corrupted images only: 100 images (one per sample)
  - No clean labels provided

Training data

- Train / validation dataset:
  - X is provided

```
train
├── L1_0009994b978735c76dba161531a98aed.npy
├── L1_00134bc9f89efb9f09bdb45a4093c091.npy
├── L1_00258fcc3b10a635d35c814c1a4f0e07.npy
├── L1_0038174cefc3df4be1ad04962ee2809d.npy
├── L1_0044a9a84b2b487a6eaa2f694a4f2153.npy
├── L1_004df5a0c62d4850720c00a5d2afb40a.npy
├── L1_008e7acbd8b621d496211b723572de5c.npy
├── L1_00d20402369a67e1a663c812f6f6730b.npy
├── L1_00e52e8047b047cc5a50756a5f329412.npy
├── L1_00f4aa2aff02c444df77448b60942d63.npy
```

```
val
├── L1_04e524d382fc4f1aecdf6bbf39bd5b0f.npy
├── L1_0aceacb0b3bbb83e38dd3f314b5b61b8.npy
├── L1_0b01fddd63d7c84949f7439279d7bd21.npy
├── L1_0b8e81c4fb1673f558be432fc1e3360f.npy
├── L1_0bfd3fe207e38d812a4e86de28364d95.npy
├── L1_0d8fa2fe52b7723f0577f914270b4489.npy
├── L1_0e5cf227eb9e5dab0135ddeb53462413.npy
```



  - Convolution & add noise can be performed via simulator

Test

- Test dataset:
  - Corrupted images only: 100 images (one per sample)
  - No clean labels provided
  - Test code is provided with encrypted evaluator
  - Run test code with your test data on the Colab

```
evaluator = _evaluate.Evaluator(
    data_root=str(ROOT / "evaluation/test"), label_root=str(ROOT / "evaluation/label")
)
evaluator()
```

```
2025-08-11 17:40:17 [INFO] Evaluator initialized with data root: /mnt/d/list/samsung_DS/2025_summer/실습프로젝트/evaluation/
2025-08-11 17:40:18 [INFO] Loaded 100 data files and 100 label files.
2025-08-11 17:40:18 [INFO] Average pSNR: 119.191, std: 1.200
2025-08-11 17:40:18 [INFO] Average SSIM: 1.000, std: 0.000
```

Task

- Use the forward model to synthesize training pairs
- Construct a pipeline to restore corrupted images into clean images
- You are free to explore any architectures or algorithms
    - Classical image processing
    - End-to-end approach (corrupted → clean)
    - Two-stage approach (denoising + deconvolution)
    - Self-supervised or label-free strategies (recommended for realism)
    - Etc.
- Requirements:
    - For each clean image, you may generate up to 6 dipole kernel orientations.
    - For each orientation, you may generate up to 2 noisy images.
    - Do not exceed 12 corrupted images per clean image (6 orientations × 2 noises)

Evaluation

- Metric (on test dataset)
  - On test dataset
  - PSNR, SSIM

- Presentation
  - Explain your overall pipeline clearly
  - Show example restoration results (before/after).
  - Justify why you chose your specific method.

- Label-free pipelines will receive partial bonus consideration.