Object detection on the runway with Stereo-RCNN

CS470 Introduction to Artificial Intelligence

Team 32 (Stereo-RCNN)
Ji il Park, Hyunyong Jeon,
Leon Thormeyer

2019. 12. 2. (Mon)

Mechatronics, Systems and Control

KAIST

Contents

Introduction

Why object detection on the runway ?

KAIST Mechatronics, Systems and Control

❖ Snowplow vehicle need 3D object detection ability

- **Runway snowplow of each country**

Korea

Russia

Germany

- Runway snowplows need to <u>remove stacked snow when it snows.</u>

  It means that it is difficult to use lidar that is vulnerable to falling snow.

- Therefore, we need to <u>detect the object with vision camera.</u>

  * Radar reduces detection range by more than 50% when snowing.

  * Snow filtering lidar data is also reduced by more than 50%.

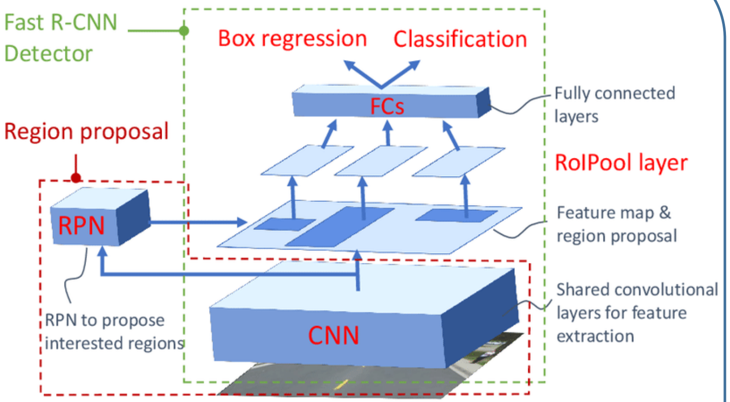
- Just in case, we need the 3D object detection ability by using just vision camera.

  → For these reason, we <u>decide to find 3D object detection model based on stereo vision camera</u>.

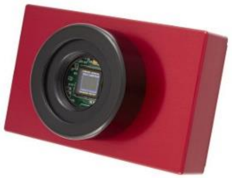
[1] https://github.com/HKUST-Aerial-Robotics/Stereo-RCNN.

What is Stereo-RCNN ?

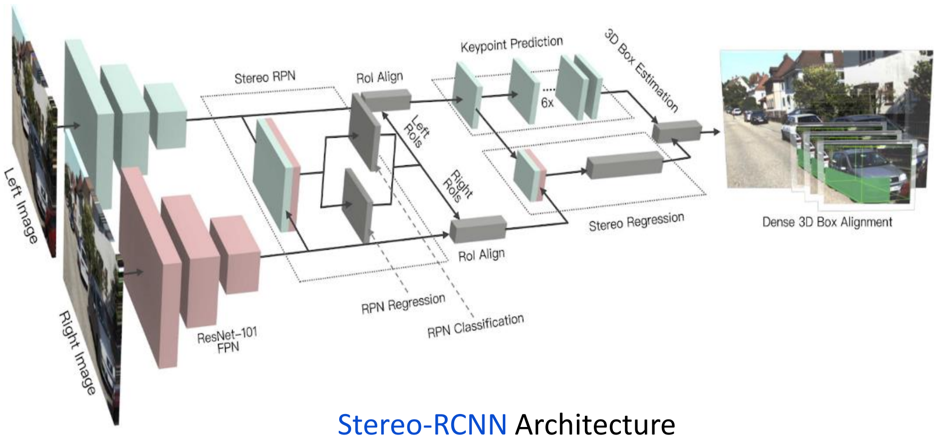
❖ Network architecture of Stereo-RCNN



Faster-RCNN Architecture



Faster-RCNN use mono image
So only make 2D bounding box



Stereo-RCNN Architecture

Stereo-RCNN can infer
depth of multi objects so
can make 3D bounding box

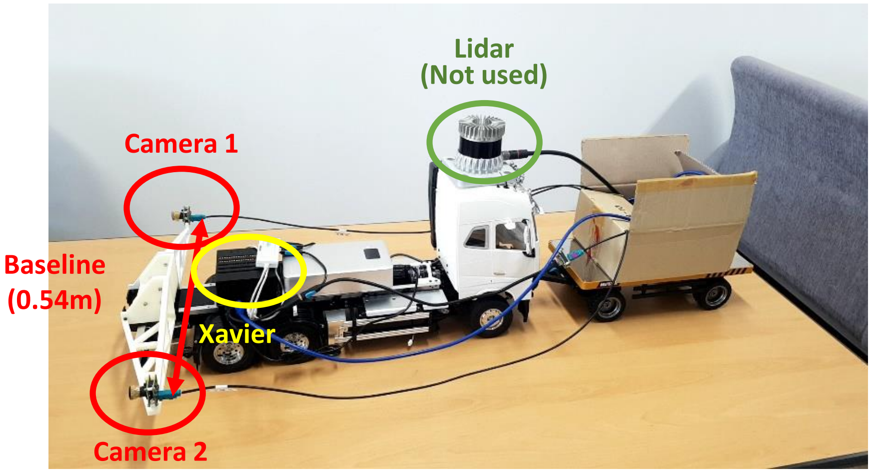
- This algorithm extends Faster-RCNN for stereo inputs to simultaneously detect 3D bounding boxes by using the left and right images.
- This algorithm is essential for autonomous driving with vision cameras without lidar.

[1] https://github.com/HKUST-Aerial-Robotics/Stereo-RCNN.

Experimental Design

Hardware setup

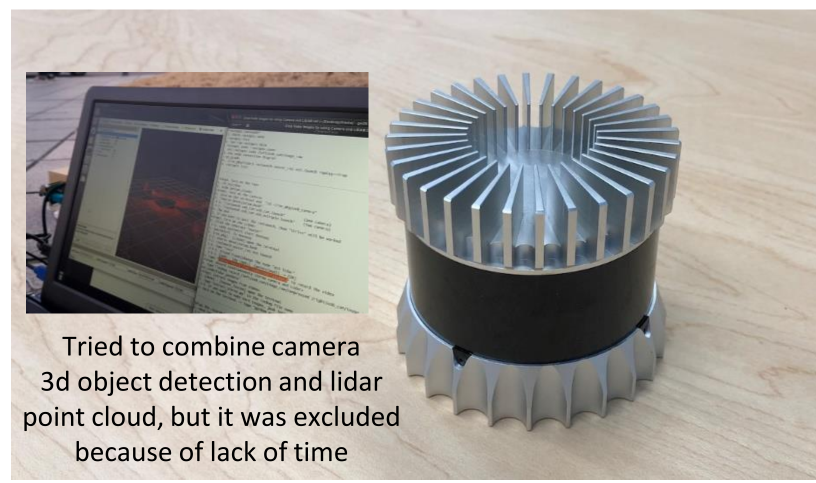
❖ **Sensor(Stereo camera + LiDAR) mounted RC car model for Object Detection**



RC car model for autonomous driving

Nvidia Jetson AGX Xavier Dev. Kit
(GTX 1060 ti grade performance)

NileCAM30_USB camera for stereo vision (2set)



Ouster 64-channel LiDAR

Algorithm modification

❖ Try #1

- Resnet algorithm improvements.

  → Since the Resnet 152 model is already applied, performance improvement is limited.

❖ Try #2

- Apply other models.

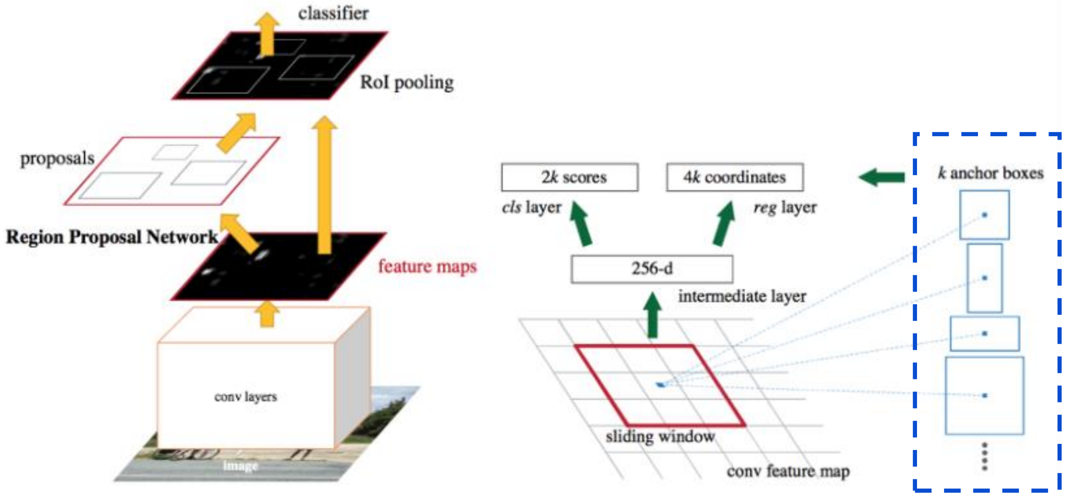
  → Applied Resnet+Google inception known for good performance but rather can't detect the object (Unknown reason)

  → VGG model performs poorly compared to Resnet, as is known in several papers.

❖ Try #3

- Decided to modify RPN part of Resnet.

  → Resize the anchor shape to fit objects on the runway such as air plane.



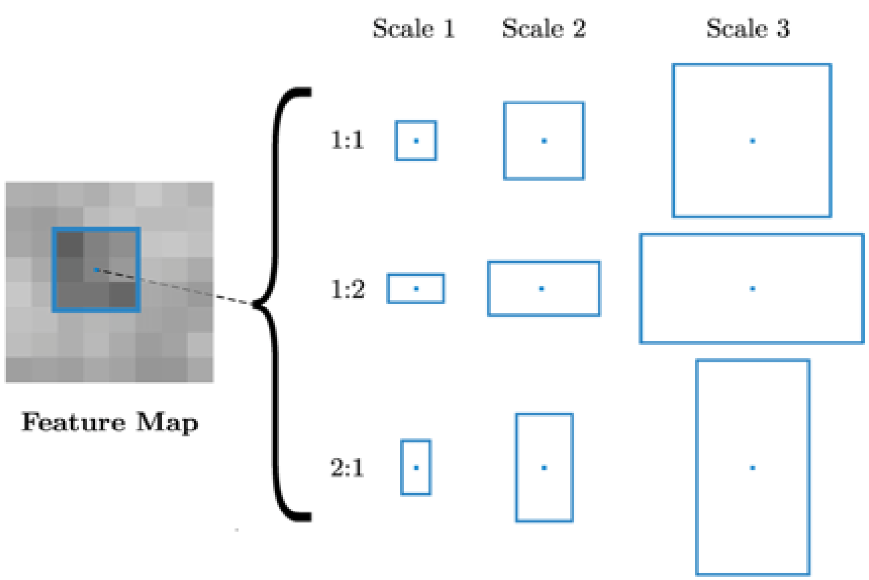
Algorithm modification

❖ RPN modification

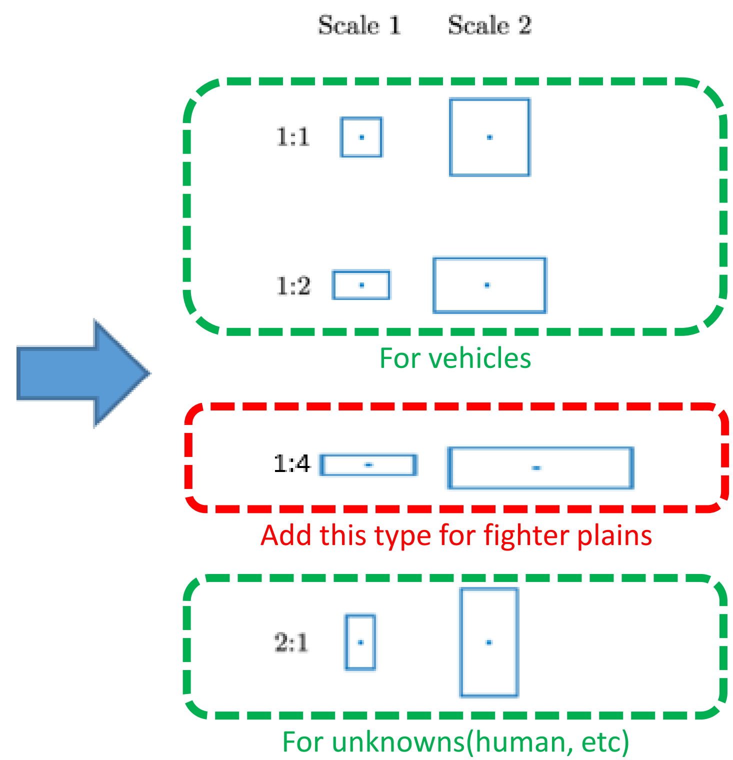
※(Assuming) 1. Operating environment : Airplane runway
2. Expected target object : Air plains, vehicles, and unknowns

- Need to change the appropriate anchor size for the operating environment.
- The object to be detected in the runway environment are mainly lateral wide air planes. As a result, we added horizontally wide anchors to identify airplanes. (1:4)

- Increasing the ratio to four types may exceed the GPU processing memory, so the scale is reduced to two types in consideration of this. (change from 3-scale to 2-scale)



Total 9 anchor Boxes



Total 8 anchor Boxes

Algorithm modification

❖ Modification of Stereo camera calibration parameter (Intrinsic & extrinsic)

- Baseline : 54cm, Checker board(Grid Rows 6, Grid Cols 8, Cell size 26mm)

Left camera

Right camera

Left camera

Right camera

- Camera parameters

(Left camera) rms = 0.148793, fx = 767.646362, fy = 769.060456, cx = 669.203517, cy = 388.021517,
k1 = -0.249038, k2 = 0.068186, p1 = 0.001469, p2 = -0.003465
(Right camera)rms = 0.128235, fx = 726.518247, fy = 727.059139, cx = 636.526418, cy = 310.456766,
k1 = -0.2, 16319, k2 = 0.057398, p1 = 0.004065, p2 = 0.006851

▶ focal length(fx, fy), principal point(cx, cy), radial distortion(k1, k2), tangential distortion(p1, p2)

- Callibaration matrix

('Intrinsic_mtx_1', array([[442.7…,]])), (‘dist_1', array([[-0.203…, …]])), ('Intrinsic_mtx_2', array([[3.839… ,]]), ('dist_2', array([[-8.568152…, …] …])), ('R', array([[8.37752308e-01, …], …])), ('T', array([[-24.72037826], …])), ('E', array([[2.02892314, …], …])), ('F', array([[1.54412202e-07, …], …]))

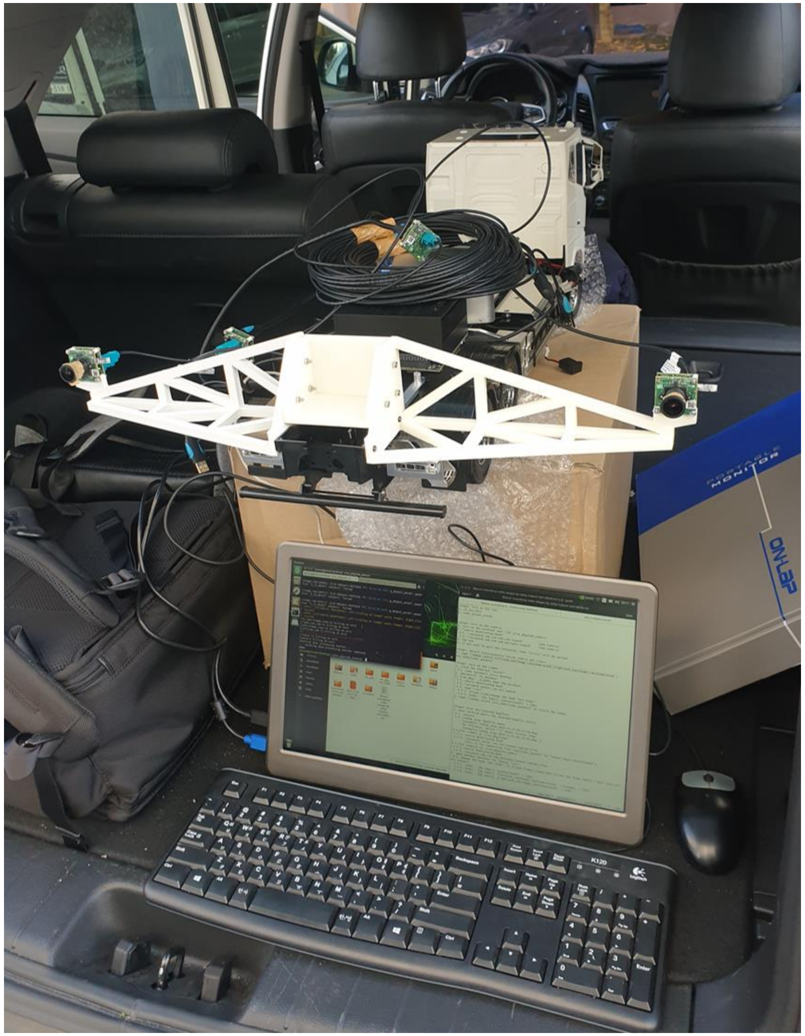
▶camera matrix(Intrinsic_mtx_1, 2), distortion matrix(dist_1, 2),
rotation matrix(R), translation matrix(T), essential matrix(E), fundamental matrix(F)

※ Intrinsic Parameters: Camera parameters that are internal and fixed to a particular camera/digitization setup.
※ Extrinsic Parameters: Camera parameters that are external to the camera and may change with respect to the world frame.

Test & Results

Data acquisition

❖ Real test driving



Outdoor test preparation

- 1st test (11.27 / KAIST Front Door) : not suitable images for testing




* Problems : ① different view angle between test & training images
② Background that makes vehicle identification difficult

- 2nd test (11.30 / Habit @) : Good test image




* Similar to the training image (vehicle shape, view angle, etc…)

Performance of Object detection (1/2)

❖ Simulation Result (Hanbit apartment test images)

- **Original algorithm**

- **RPN change algorithm**

Image #122

Image #137

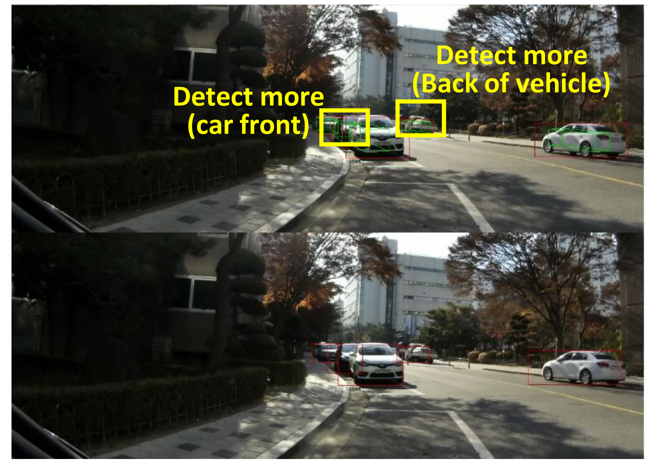
Performance of Object detection (2/2)

❖ Simulation Result (Hanbit apartment test images)

- **Original algorithm**

Bird eye view

- **RPN change algorithm**



Bird eye view

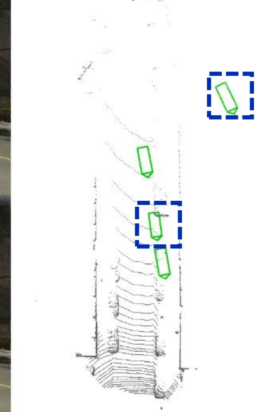


Image #354

Bird eye view

Bird eye view

Image #370

Conclusions & Further work

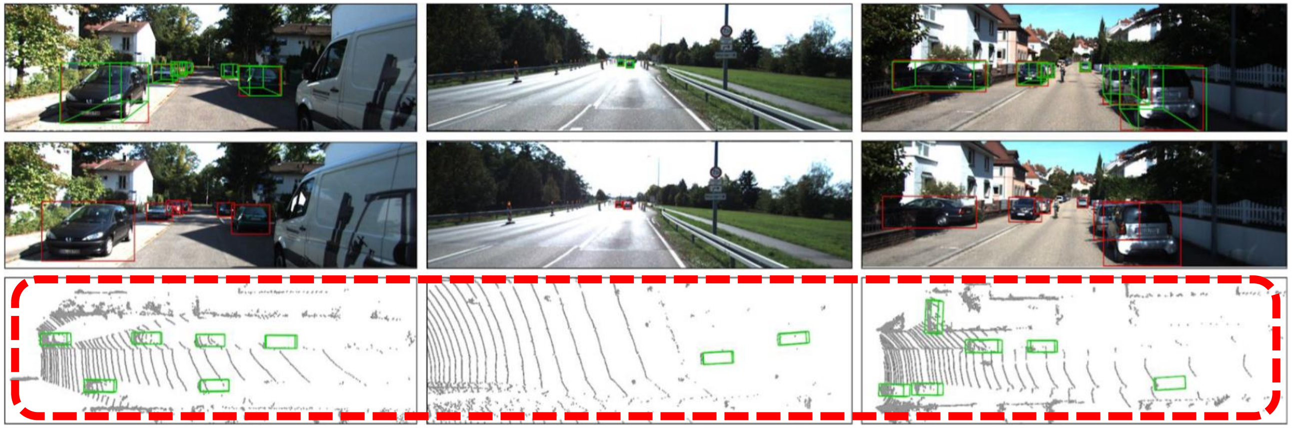
Conclusions & further work

Conclusions

- Better object detection performance despite reduced number of RPNs (9 → 8)
- (Image #122, #137, #345) Detect better when vehicles are even clustered
- (Image # 137, #345, #370) Find more objects in the distance
- In conclusion, Using RPNs that is appropriate for your environment, rather than a large number of RPNs, will perform better on object

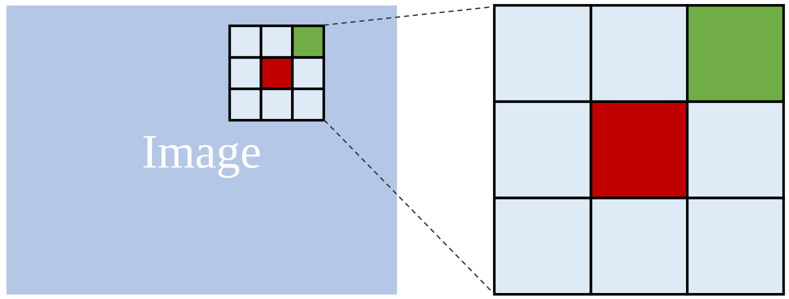
Further work

- Improved with robust stereo vision algorithm even in Extreme env(rain, snow, and night)
- Create the stereo-camera version of YOLO, and SSD algorithm
- Since the collected data doesn't contain noise, it is considered that applying the filtering process will be meaningless to confirm the our algorithm. So, implement later
- Try to get test results with bird eye view by extracting the LiDAR point-cloud data



Q & A

Appendix #1. Apply Hampel filter



: Median of window $M(x,y)$

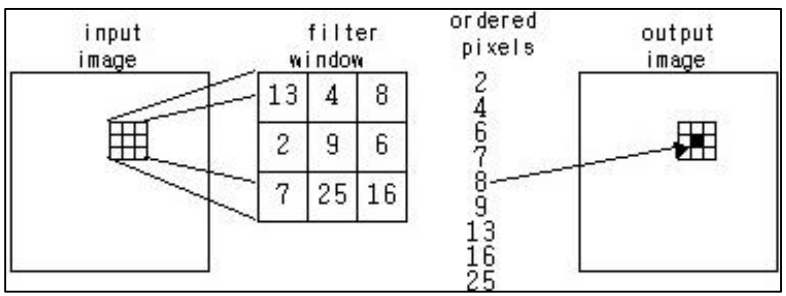
: Center of window $I(x,y)$

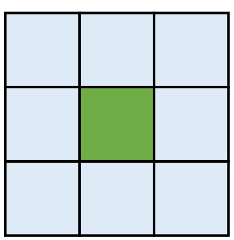
K : Half window size

❖ Median filter

1. Filter Response

$$I_f(x,y) = M(x,y)$$





Always replaced by median value of window.

❖ Hampel filter

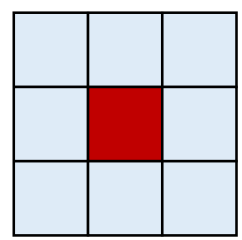
1. MAD (Median Absolute Difference) scale estimate

$$S(x,y) = 1.4826 \times median_{i,j \in [-K,K]} \left| I(x+i, y+j) - M(x,y) \right|$$

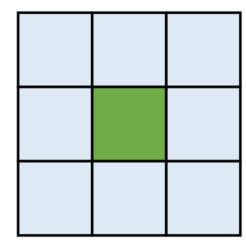
2. Filter Response

$$I_f(x,y) = \begin{cases} I(x,y) & \left| I(x+i,y+j) - M(x,y) \right| \le t \cdot S(x,y) \\ M(x,y) & \left| I(x+i,y+j) - M(x,y) \right| > t \cdot S(x,y) \end{cases}$$

- Two tuning parameters: ***K* (filter size), *t* (error bound)**



Center of window is **NOT** outlier: Preserve original value



Center of window **IS** outlier: Replace to median value

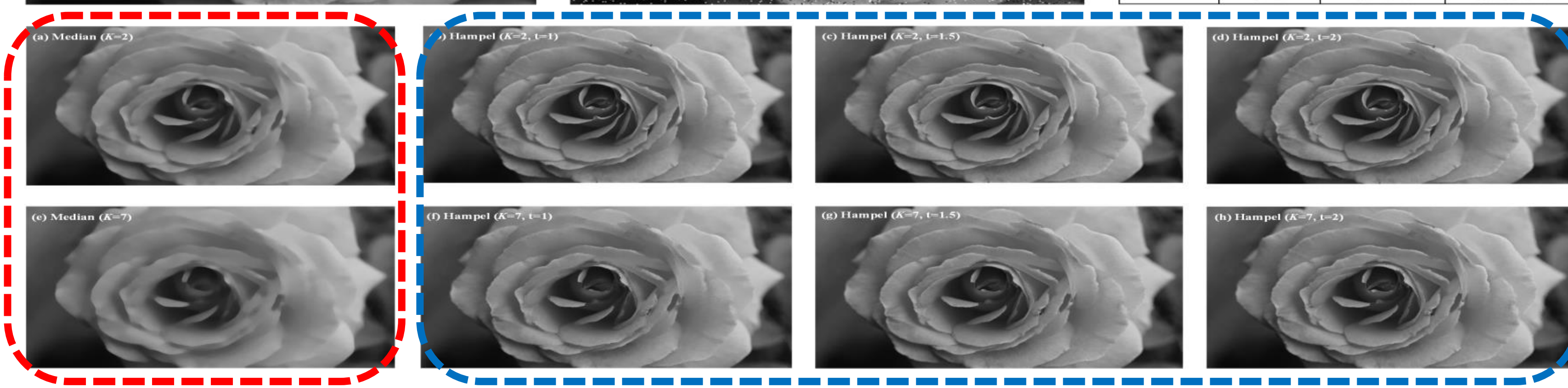
Appendix #2. Apply Hampel filter

❖ Benefit of Hampel filter

- Excellent for preserving the original image compared to other image filters like media filter which is a representative image filter.
- Easy simulation through parameter tunning. In particular, the filtering intensity can be adjusted by adjusting the error bound value.
- If the error bound value is set to 0, this filter can be used as a median filter.
- By applying the weighted concept, it can be used as a weighted hampel filter. This can minimize the collapse of the boundary of an object in the image.

❖ Result of Hampel filter on gray-scale image

Window size	Error bound	SSIM	MSE
K=2	t=0	0.97573	0.010251
	t=1	0.99041	0.0059517
	t=1.5	0.99622	0.0040323
	t=2	0.99704	0.0040253
K=7	t=0	0.8812	0.033688
	t=1	0.92974	0.025837
	t=1.5	0.95891	0.021425
	t=2	0.96986	0.019208



Median filtered image

Hampel filtered image