

© Copyright 2021

Shinaphadh Damrongsiri

Development of Particle Identification Technique for Particle Tracking
Velocimetry Application in the Presence of Image Noise

Shinaphadh Damrongsiri

A thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Aeronautics & Astronautics

University of Washington

2021

Committee:

Dana Dabiri

James Hermanson

Program Authorized to Offer Degree:

Aeronautics & Astronautics

University of Washington

Abstract

**Development of Particle Identification Technique for Particle Tracking
Velocimetry Application in the Presence of Image Noise**

Shinaphadh Damrongsiri

Chair of the Supervisory Committee:
Professor Dana Dabiri
William E. Boeing Department of Aeronautics & Astronautics

For Particle Tracking Velocimetry (PTV), the presence of digital image noise deteriorates both the particle localization and identification performance. In this thesis, a proposed workflow combines a state-of-the-art deep-learning based denoising architecture, U-Net image segmentation technique, and particle reconstruction through linear model inversion. A number of simulation tests under different noise conditions and particle density using synthetically generated images are performed in order to evaluate the performance improvement against traditional methods. At the particle density of 0.10 particle per pixel and 5 percent image noise, the proposed workflow reduces the Mean Localization Error by 24 percent compared to clean image. The workflow requires no prior knowledge in noise level nor the particle density. Also, the Gaussian residual image noise optimization for particle reconstruction technique is proposed for non-overlapping particle image in presence of image noise.

TABLE OF CONTENTS

List of Figures	iv
List of Tables	viii
Chapter 1. Introduction	11
1.1 General Background	11
1.2 Research Objective	12
1.3 Thesis Outline	12
Chapter 2. Theoretical Background	13
2.1 Particle Image Velocimetry (PIV)	13
2.2 Particle tracking velocimetry	15
2.3 Particle Image Identification	17
2.3.1 Modified Cascade Cross-Correlation (MCCM) approach	17
2.3.2 Particle Identification through Point Spread Function (PSF) model inversion	18
2.4 Localization error in particle location	20
2.5 Image segmentation techniques	24
2.5.1 Threshold Binarization	24
2.5.2 U-Net Convolutional Network image segmentation	27
2.6 General Background on Image noise	29
2.7 Removal of Image noise	30
2.7.1 Image Denoising Introduction	30
2.7.2 Deep Convolutional Neural Network for Image Denoising (DnCNN)	33

2.7.3	Residual Dense U-Net Network for denoising (RDUNet)	34
2.7.4	Comparison of modern denoising technique	37
Chapter 3. Methodology		39
3.1	Proposed particle image identification algorithm	39
3.1.1	Split Image into 32 by 32 pixel.....	41
3.1.2	Particle Image Segmentation using U-Net.....	42
3.1.3	Particle image denoising using Residual Dense Neural Network (RDUNet).....	46
3.1.4	Particle image reconstruction from particle image through model inversion.....	51
3.1.5	Particle image segmentation using U-Net.....	52
3.1.6	Reliability improvement using distance transform.	53
3.1.7	Reliability improvement using density based spatial clustering.....	55
3.2	Synthetic Image Generation.....	58
3.3	Particle Identification Performance Metrics	59
3.4	Residual noise profile optimization using iterative particle reconstruction technique .	60
3.4.1	Reconstruction procedure	61
3.4.2	Image noise comparison: Statistical Model selection.....	62
3.4.3	Cost Function	64
3.5	Random Particle Radius Identification Test	65
Chapter 4. Results		66
4.1	Critical Parameters Selection.....	66
4.1.1	Distance Transform rejection threshold.....	66
4.1.2	Density Based Spatial Clustering thresholds	67

4.2	Numerical Results	69
4.2.1	Particle Identification Performance at 5 percent Gaussian noise.....	69
4.2.2	Robustness of Particle Identification Performance at different noise level.....	71
4.2.3	Robustness of Particle Identification workflow at large image size.....	76
4.3	Computational Time	80
4.4	Particle reconstruction results	81
4.5	Random Particle Radius Results	82
4.5.1	Random size from 1 to 1.2 pixel.....	82
Chapter 5. Future Work		85
5.1	Performance drop mitigation on large image.....	85
Chapter 6. Conclusion.....		86
Bibliography		87

LIST OF FIGURES

Figure 2.1 An example of particle image. A Gaussian profile particle (Left). A Gaussian profile particle contaminated with 40% Gaussian noise (Right).	21
Figure 2.2 The 2D representation of particle image in Figure 2.1. The contaminated particle image represented in red line clearly demonstrated the effect of the Gaussian peak shift due to variation in pixel intensity.	22
Figure 2.3 Average particle location error at various Gaussian noise level [9].	23
Figure 2.4 The particle image (Left) with binary thresholding mask generated using Otsu's method (Right).	24
Figure 2.5 The noisy-segmented particle image pair using single thresholding method..	25
Figure 2.6 The comparison of Single threshold binarization and Dynamic threshold binarization [6]. It can be seen from (b) that using single thresholding would leave a large amount of particle out. In (e) for dynamic thresholding technique, a smaller number of particles are lost compared to (c)	26
Figure 2.7 Basic U-Net architecture. The arrows represent the different operations, the blue boxes represent the feature map at each layer, and the gray boxes represent the cropped feature maps from the contracting path [16].	27
Figure 2.8 An example of Semantic Segmentation of Multispectral Images Using Deep Learning (U-Net) [19]. The trained U-Net algorithm label the multispectral image through recognition of spatially distributed structural information previously learned during the training.	28
Figure 2.9 The architecture of DnCNN denoising network [26].	33
Figure 2.10 Architecture of RDUNet Neural Network [28].	35
Figure 2.11 Building blocks of the RDUNet: a) Basic components used in the building blocks of the RDUNet model, b) input and output blocks, c) densely connected denoising block used throughout the model [28].	35
Figure 2.12 Performance of different denoising methods on BSD68, Kodak24 and Set12 grayscale datasets with different noise levels [28].	37

Figure 3.1 Process flow of our proposed particle image identification algorithm.....	40
Figure 3.2 The original image of size 160 by 160 is spitted into smaller image patch of size 32 by 32 and individually processed through the proposed workflow.	41
Figure 3.3 The plot representation of U-Net model for particle image segmentation task.	42
Figure 3.4 Randomly selected Noisy-Label training image pair used for U-Net algorithm training. Left: Noisy image as input for U-Net. Right: Binary label (overlayed on noisy image for illustration purpose only).....	44
Figure 3.5 Randomly selected image pairs for training. In this image pair, the N_{ppp} 0.10 and 5% noise. Left: Clean particle image. Right: Noisy particle image with 5% of added Gaussian noise. Images are colorized based on their pixel value for illustration purpose.	47
Figure 3.6 The sparse (right) particle representation of the particle image (left).	51
Figure 3.7 Visual illustration of background False Positive (FP) particle rejection. Left: Unprocessed detections from PSF. Mid: U-Net binary mask indicate the region where particles are present. Right: Output of the algorithm.....	52
Figure 3.8 Left: The distance transform of binary mask from U-Net is scaled up 20 times to increase resolution. Right: The clean-up detection by rejecting any detection that below the γ_{optim} threshold.	54
Figure 3.9 Example of clustering of points using DBSCAN algorithm [36],.....	55
Figure 3.10 The detections of particle in noisy image through PSF method that already refined by U-Net and distance transform technique. Each number corresponding to specific color represent the clustering of points using DBSCAN algorithm. The number less than 0 indicate outliers, whereas the number greater than 0 indicate multiple detection of the same particle. Hence the centroid of each cluster is calculated. The red cross indicates the ground truth position of each particle.....	56
Figure 3.11 Top: Zoom-in version of a particle image. The centroid of cluster 5, 6, and 7 are computed. Bottom: The centroid of cluster 14, 15, 18 19 are computed.....	57
Figure 3.12 Proposed residual noise optimization workflow for non-overlapping particle image.	60
Figure 3.13 The binary blob containing particle. Every single pixel in each blob is used to calculate a centroid corresponding to a particle.....	61

Figure 3.14 The image of 5 percent Gaussian noise. The blue circle separates two ROI, the inside of circle and the outside of circle for comparison.....	63
Figure 3.15 The value-pair of each statistical tests.....	63
Figure 4.1 The plot of yield and reliability performance matrices while varying α parameters. The vertical magenta line indicates the optimum value of α chosen for the workflow.	66
Figure 4.2 The plot of yield and reliability performance matrices while varying γ_{dbscan} parameters. The vertical magenta line indicates the optimum value of γ_{dbscan} chosen.	67
Figure 4.3 Comparison between Clean image using PSF (Black), Noisy image using PSF (Red), Noisy image denoised using DnCNN (Magenta), and Noisy image denoised with RDUNet (Blue) in terms of Mean localization error, yield, and reliability matrices. Regardless of denoising techniques, all methods subjected to our proposed False Positive (FP) rejection workflow.	69
Figure 4.4 The Mean Localization Error at different noise level from 1 to 4 percent.	71
Figure 4.5 The yield at different noise level from 1 to 4 percent Gaussian noise.	72
Figure 4.6 The reliability at different noise level from 1 to 4 percent Gaussian noise.	72
Figure 4.7 The detection results for the 160 by 160 pixels image at $N_{ppp} = 0.10$ and 5% Gaussian noise. The ground truth location (red cross) and detections (blue circle) are superimposed over the particle image.....	76
Figure 4.8 The detection of proposed workflow evaluated in terms of True Positive (TP), False Positive (FP) and False Negative (FN).	77
Figure 4.9 The regions where individual image patches are merged indicated in red rectangular box. The number of False Positive detections (FP, yellow) in the indicated regions clearly increased.	78
Figure 4.10 The 160 by 160 pixel particle image performance (Green) in comparison to small image patch 32 by 32 pixel.	79
Figure 4.11 The plot of elapsed time for each image patch (32 by 32 pixels) against the particle density N_{ppp}	80
Figure 4.12 The boxplot representation of particle location error of using only 2D Gaussian function fit (Red), denoise image using DnCNN then 2D Gaussian fit (Green), our proposed	

iterative particle reconstruction and residual noise comparison (Blue). The number associated with each data refers to the Gaussian noise level in percent.	81
Figure 4.13 The localization performance for varying radius particle (1 to 1.2 pixel).....	82
Figure 4.14 The Yield and Reliability performance of varying particle radius (1 to 1.2 pixel).	83
Figure 4.15 The Yield and Reliability performance of varying particle radius (0.6 to 1.4 pixel).	84

LIST OF TABLES

Table 3.1. Critical Parameters for U-Net model creation	43
Table 3.2. Image Patch Generation for U-Net training.....	43
Table 3.3. U-Net Image Segmentation Training Parameters	45
Table 3.4. Image Patch Generation for RDUNet training	48
Table 3.5. RDUNet Training Parameters.....	49
Table 4.6. The Mean Localization Error at different noise level for each technique.	73
Table 4.7. The Yield at different noise level for each technique.....	74
Table 4.8. The Reliability at different noise level for each technique.....	75
Table 4.9. Particle Identification and Localization Performance for large-scale image...	78

ACKNOWLEDGEMENTS

I would like to thank my research supervisor, Professor Dana Dabiri, whose expertise was invaluable in the investigation of highly accurate particle tracking algorithms. Thanks to his great advice in research feedback and methodology, I could accomplish this great piece of work.

I would like to thank Professor James Hermanson for your time and effort to serving on my reading committee and final thesis presentation. Your insightful suggestions and ideas pushed me to sharpen my ideas and brought this work to a higher level.

I would like to thank Yi-Lin Liu for wonderful collaboration in both experimental and synthetic image simulation work in AERB 214 laboratory.

Last but not least, I could not say thank you enough to my family who encourage me and always be by my side.

DEDICATION

To my family and friends.

Chapter 1. INTRODUCTION

1.1 GENERAL BACKGROUND

The behavior of fluid flow in nature has long been a fascinating topic for humans to understand in terms of its behavior and its interaction with any objects in its path. In the simplest terms, some might believe that the term “Fluid” refers to only liquids e.g., water, human blood, but the term ‘fluid flow’ includes everything that can move along the direction of stream. Unlike solids, where the molecules are tightly packed together and hardly move, and thus, their shape does not change. Fluid, on the other hand, has no fixed shape and can deform easily due to changing external pressure and/or shear. This distinctive property of fluid means that it can travel through a path where solids cannot allow nature to use that, for example to advantage of transporting oxygen and energy in our blood stream. Owing to the fact that the fluids are responsible for much of things that affect our life and society, a thorough understanding of what is happening inside the flow of fluid is a very crucial step to enables more precise and accurate engineering applications.

Flow visualization techniques has long been a primary way used to analyze and enables humans to visually inspect what is happening inside the flow of fluid. Particle Tracking Velocimetry (PTV) is among the greatest achievement of engineering society in modern fluid mechanics [1]. In principle, the motion of fluid flow is captured through reflective particle suspended in fluid using a sequence of image through time [2].

It is an of the utmost importance to identify and locate particles from images, because the derivation of the velocity field is calculated from displacement of particle between an image pair,

taken with a specific time delay Even the slightest uncertainty of particle locations will lead to much greater degree of error.

1.2 RESEARCH OBJECTIVE

In this thesis, we investigate various approaches that aim to identify as much particles as possible while maintaining the highest possible location accuracy from a noisy image. The widely used particle identification and localization performance matrices are used to generate the results, therefore the comparison can be made to evaluate performance of the workflow among other methods.

1.3 THESIS OUTLINE

The thesis is organized as follows. In Chapter 2, the general background knowledge of the algorithm and software that is used throughout the thesis are described. In Chapter 3, the methodology for each experiment and critical parameters are outlined. In Chapter 4, the results for each approach are described and compared where possible. In Chapter 5, the possible improvement for current work is suggested. In Chapter 6, concluding remarks are stated.

Chapter 2. THEORETICAL BACKGROUND

This Chapter 2 describe some of the well-known techniques used to track particles inside the fluid flow. Later in this chapter, the effects of image noise on particle identification accuracy are described and investigated. Also, several approach that are effective in removing error and enhancing the particle identification performance are described and benchmarked.

2.1 PARTICLE IMAGE VELOCIMETRY (PIV)

It is obvious that we cannot observe the flow of water or other fluid by naked eyes, as different type of the tracer particle is put into fluids flow to extract motion generated by the fluid. The tracer particle must be sufficiently small such that when put into fluid they closely resemble the actual movement of fluid. After that, the flow velocities can be extracted from velocities of tracer particles. Then illumination source is used to luminate tracer particle while high speed camera is capturing particle illumination.

One of the breakthroughs in PIV technologies is the Laser Speckle Velocimetry (LSV), where laser light is used to illuminate a thin cross-sectional flow plane, reflecting reflective particle in the flow [3]. In LSV, the velocity field resolution is directly corresponding to the density of seeding particle. This LSV techniques give engineers a first important step in quantifying kinematic flow field. In the beginning, the work of identifying and tracking each individual particle pair is done manually by a human. These techniques become increasingly difficult to perform accurately for high resolution flow field data as the spatial resolution of flow field is dependent on seeding density.

However, in PIV, the spatial resolution of flow field can only limit to the mean value within the interrogation window. Whereas in PTV, flow field is calculated directly from the displacement of particle between image pair [4]. In case of turbulent flow, using mean velocity field would leave out complex flow field behavior and if we use calculated mean value from interrogation window, large amount of information will be lost due to averaging effect of PIV. For this reason, PTV is more advantageous than PIV in the application that the velocity gradient is large [5].

2.2 PARTICLE TRACKING VELOCIMETRY

In Particle Tracking Velocimetry (PTV), the first step is to identify and extract pixel location where particles are present. This step is known as particle localization step. The particle localization step, after particles are located, these locations are to be linked across image frames to establish particle trajectory. In experimental images, image noises are usually present due to many sources of interference. In order to separate particle from noise, *Ohmi et al.* proposed to use image thresholding to eliminate background noise below specified threshold [6].

Up until now, there is the beliefs that particle trajectory linking is more important than particle localization in terms of final accuracy. This imbalance in development techniques resulted in a significantly less development in particle localization compared to particle trajectory establishment [7].

In PTV, the source of errors can be identified from 2 sources, particle acceleration error and particle location measurement error [8]. The Particle acceleration error is caused when particles are displaced between image frames. The particle acceleration error cannot be determined whether particles are moving in a straight line or in circular motion. This error could be mitigated by capturing particle image at a higher frame rate. In terms of particle location error, this type of error arises from the location uncertainty of particle. The location uncertainty is caused by 2 main different reasons: random image noise, which caused by digital camera sensor and the size of pixel itself [8].

In terms of particle location uncertainty, it can be categorized into two main sources: the fixed size of camera pixel and random noise that arise from camera sensor [8]. To illustrate the effect of inaccuracies of particle location on velocity field of fluid, an example of simple two-frame tracking method, the velocity uncertainty is calculated in Equation (2.1) [8].

$$(\delta V)^2 = \frac{(\delta x_{j+1})^2 + (\delta x_j)^2}{(\Delta t)^2} \quad (2.1)$$

Where δV defines the particle uncertainties associated with location uncertainties pair δx_{j+1} and δx_j taken at time difference Δt apart.

Moreover, when particles are located in close proximity to each other, the peaks of overlapping particle can be very difficult to locate [9].

2.3 PARTICLE IMAGE IDENTIFICATION

The process of particle image identification is to extract the particle location, number of particles, and various parameters from particle image.

2.3.1 *Modified Cascade Cross-Correlation (MCCM) approach*

Particle image identification is a process where the location of each individual pixel is determined. After initial particle locations are identified, the method of determining exact particle coordinate depends on whether the particles are overlapped or not. In case of non-overlapping particle, *Takehara & Etoh* proposed a method named “Gaussian Mask Correlation” that use a “model image” assuming that the particles have intensity following Gaussian profile [9]. Then a cross-correlation operation are performed across the image, yielding the peak location of each particle image that closely match the model image [9].

To maximize spatial resolution particle tracking velocimetry, highest number of particle density is always desirable. But, increasing the particle too much can resulted in overlapped particles which forbid proper particle centroid calculation [10]. When particles are separated at enough distance, the performance of particle identification using Gaussian Mask Correlation is adequate [9]. However, when the particle is not separated at enough distance, the densely packed particles will be resulted in overlapping particle image. The simulations performed by *Marxen et al.* shows that if the separation distance between particles approaches the size of particle diameter, the particle location error becomes prominent [10]. In order to overcome this, a modified cross-correlation method (MCCM) was proposed to find particle peaks through the maximum correlation between a particle representation model and particle image using 5 point Gaussian fit [9].

2.3.2 Particle Identification through Point Spread Function (PSF) model inversion

According to the tomographic model proposed by *Elsinga et al.*, the pixel intensity of particle image captured by camera is a result of the integration of light source, that are spatially distributed in the volume through the aperture diffraction [11]. With the assumption of the pixel intensity through this light integration model, *Champagnat et al.* proposed that the relationship between spatially distributed light source and the pixel intensity value can be put to use and recover accurate particle position from known pixel intensity [12]. The optical function proposed is called the Point Spread Function (PSF). The Point Spread Function can be described as the linear model coupler between the 3D representation of particle to 2D particle image [12].

The intensity model for P particles proposed by *Champagnat et al.* is described in Equation 2.1:

$$I(x) = \sum_{p=1}^P E_p h(x - F(X_p)) \quad (2.1)$$

For simplicity, it is acceptable to assume the PSF model as a simple Gaussian distribution with radius equals to σ_{psf} [12]. The PSF equation is shown in Equation 2.3.

$$h(x, y) = \frac{1}{4} \left(\operatorname{erf} \left(\frac{x+0.5}{\sqrt{2} \sigma_{psf}} \right) \right) - \frac{1}{4} \left(\operatorname{erf} \left(\frac{x-0.5}{\sqrt{2} \sigma_{psf}} \right) \right) \left(\operatorname{erf} \left(\frac{y+0.5}{\sqrt{2} \sigma_{psf}} \right) \right) - \frac{1}{4} \left(\operatorname{erf} \left(\frac{y-0.5}{\sqrt{2} \sigma_{psf}} \right) \right) \quad (2.2)$$

In the real image, the image data is discretized due to the camera sensor. Therefore, the model is as shown in the Equation 2.4.

$$I(k) \approx \sum_n h(k - n\Delta_{\tilde{E}}) \tilde{E}(n) \quad (2.3)$$

Where $I(k)$ is the pixel intensity that is indexed by k and $\tilde{E}$. By leveraging these relations between particle coordinate and particle image modelled by PSF, *Champagnat et al.* propose to rearrange the Equation 2.1 as the following:

$$I = W\tilde{E} \quad (2.4)$$

Where $W = h(k - n\Delta_E)$ is defined as weight matrix indexed by (k, n) [12].

The objective of this equation is to recover $\tilde{E}$ from known pixel intensity I , therefore, the non-negative least square (NNLS) is suitable for this problem [12]. The non-negative least square solves $\tilde{E}$, the sparsely distributed light source, through minimization of the Equation 2.5:

$$\text{minimize } \frac{1}{2} \|W\tilde{E} - I\|_2, W \in \mathbb{R}^{N \times M}, I \in \mathbb{R}^N \text{ subject to } \tilde{E} \geq 0 \quad (2.6)$$

That paper suggested that the accuracy of the recovered particle location highly dependent on the choice of the sub-pixel size Δ_E , the smaller the sub-pixel size, the higher the recovered particle location accuracy. Nonetheless, increasing the subdivision of pixel exponentially increase the sparsity of the matrix and hence, the memory consumption in calculation increased [12].

The paper also studied the series of parameters to evaluate the optimum value for Δ_E in terms of the highest accuracy while maintaining affordable computational time. The value of Δ_E equals to 20 was selected to be the optimum value [12].

In comparison to the MCCM particle identification technique, the PSF model inversion method achieves much better particle location accuracy than MCCM.

2.4 LOCALIZATION ERROR IN PARTICLE LOCATION

In general, the particle diameter is less than 2 pixels. Because of a relatively small particle size when comparing to the size of a pixel, the least square function fit over Gaussian intensity to locate the particle position would yield a significant systematic location error. In case that the particle image is contaminated by noise, more errors could introduce to the position. This is because both low resolutions of particle image and noise can alter the particle localization result that is obtained from Gaussian mask fit.

Furthermore, at the higher strength of image noise, there will be further increase in the location error due to shifting of the calculated peaks of Gaussian profile. Despite the recent advancement in camera hardware and sensors to improve image quality, the random disturbance in digital imaging is inevitable. Also, the variation in pixel intensity can result in confusion of particle detection algorithm where the detection does not contain any particle. The corrupted image by noise is compared with the clean image in Figure 2.1.

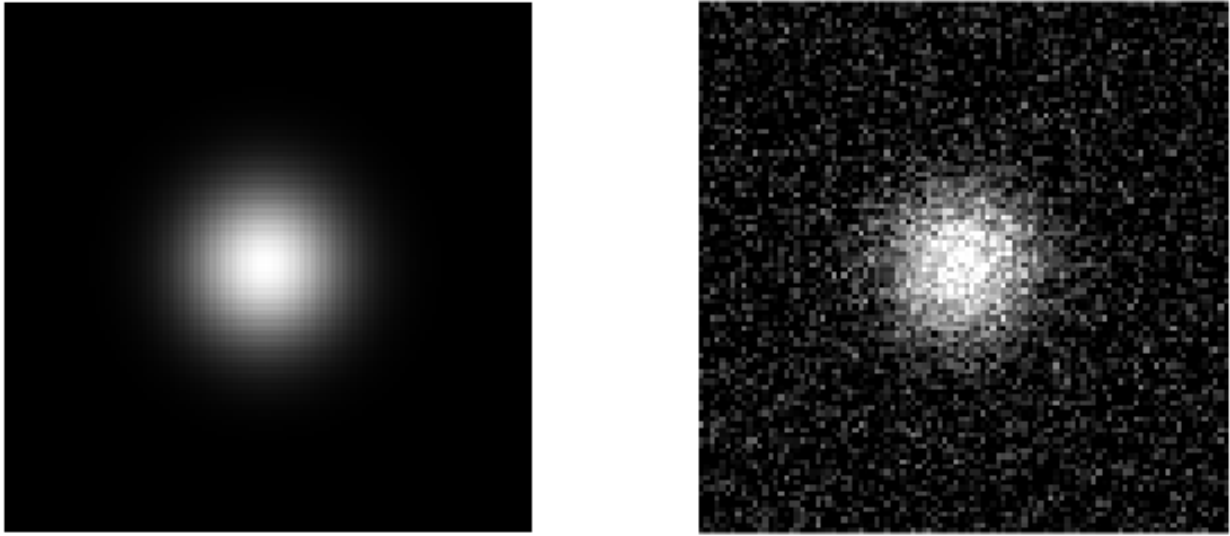


Figure 2.1 An example of particle image. A Gaussian profile particle (Left). A Gaussian profile particle contaminated with 40% Gaussian noise (Right).

In order to illustrate the distortion of Gaussian intensity due to the noise, the summation of pixel value in vertical direction is plotted against the horizontal distance from the edge. The plot is shown in Figure 2.2.

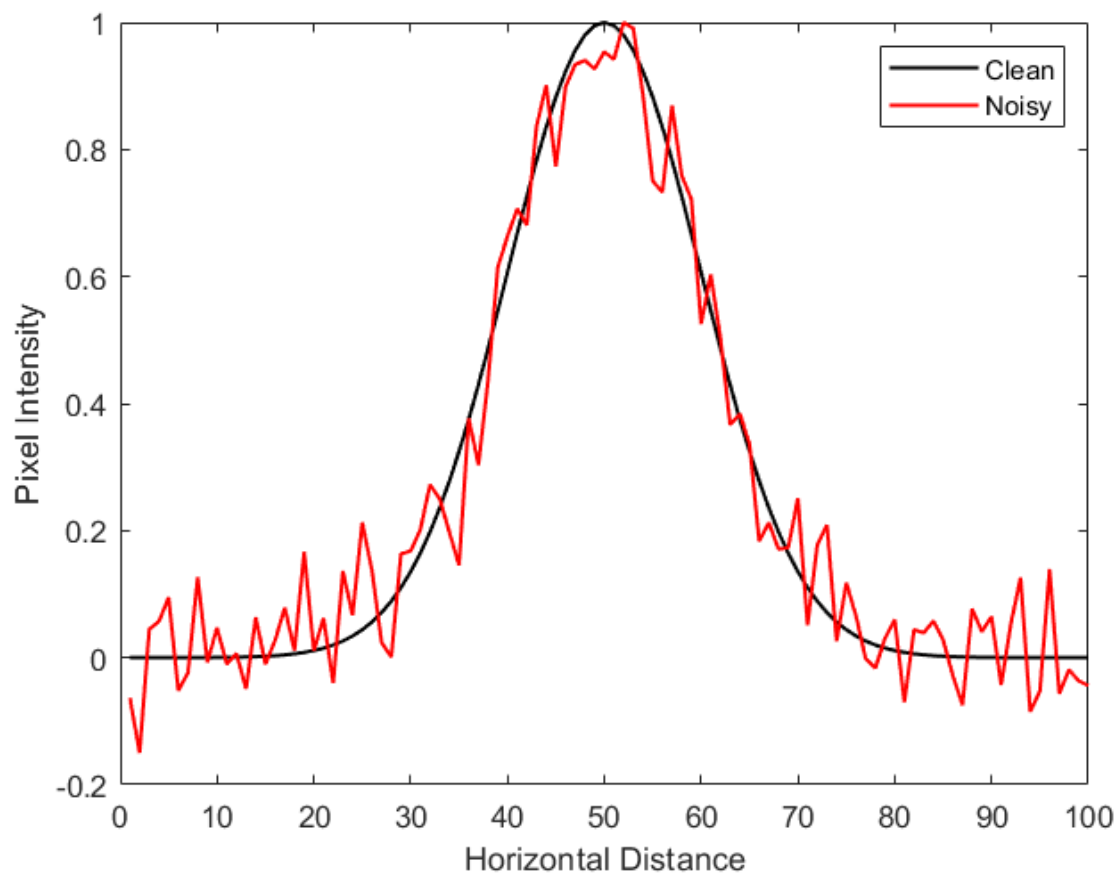


Figure 2.2 The 2D representation of particle image in Figure 2.1. The contaminated particle image represented in red line clearly demonstrated the effect of the Gaussian peak shift due to variation in pixel intensity.

According to the paper published by *Lei et al.*, the Gaussian noise was added to the particle image to replicate experimental image noise, the noisy image was processed through their proposed PTV algorithm. It can be seen in the Figure 2.3, in the perspective of Averaged Location Error (pixel) that the particle location error increase proportionally to the amount of added noise [9].

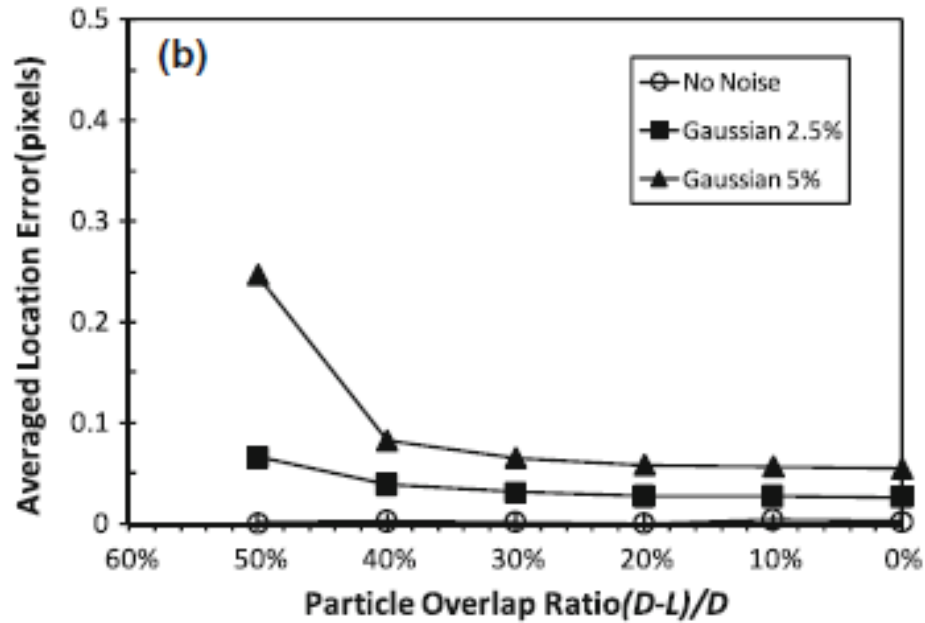


Figure 2.3 Average particle location error at various Gaussian noise level [9].

Therefore, it is very important that a suitable denoising algorithm needs to be applied to particle image in order to reduce the particle location error.

2.5 IMAGE SEGMENTATION TECHNIQUES

As far as the development in the digital camera technology, the image noise still presents in the image. There are techniques to separate particles from noisy background, these techniques are called image segmentation technique. In the case that background regions are identified, the detection in the region can be removed.

2.5.1 *Threshold Binarization*

Traditionally, a well-known approach to separate particle from background from noisy image is to apply a single threshold binarization to locate the particle peak in image. This single thresholding removes all pixel noise that are below prespecified intensity. The threshold intensity can be calculated using *Otsu's method* [13]. However, this method can generate multiple erroneous regions due to variations of pixel intensity as shown in the right half of Figure 2.4. Using the binary mask from Figure 2.4, the segmented particle image can be calculated (Figure 2.5).

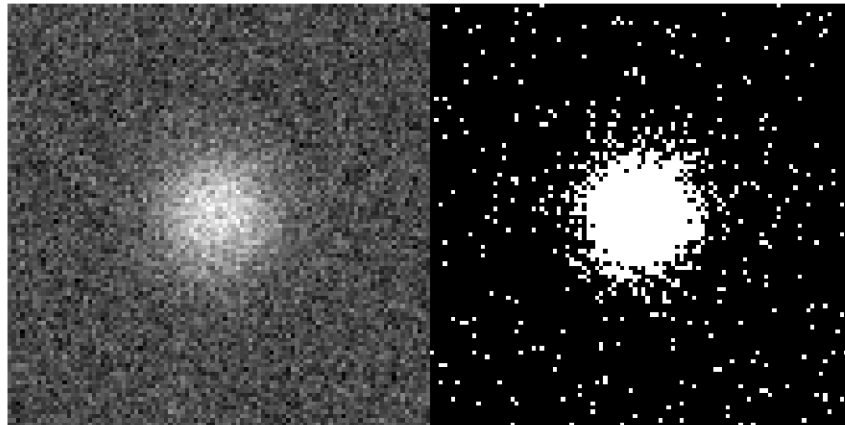


Figure 2.4 The particle image (Left) with binary thresholding mask generated using Otsu's method (Right).

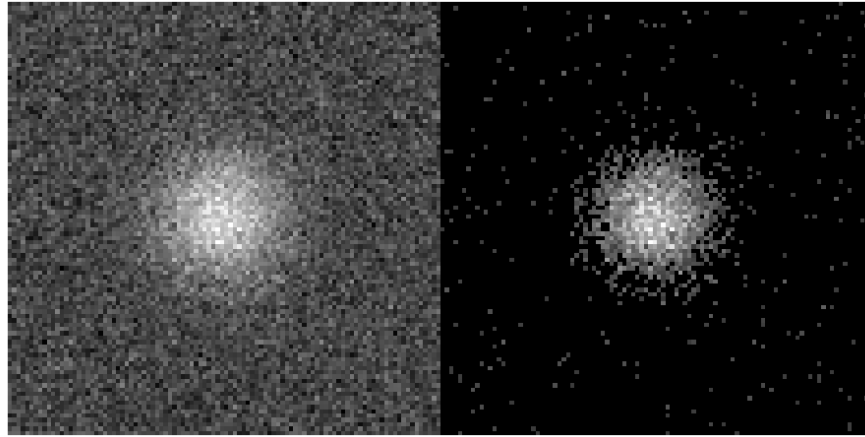


Figure 2.5 The noisy-segmented particle image pair using single thresholding method.

However, as particle density increases, the combination of various particle arrangement causes variation in the particle peaks intensity. *Ohmi and Li* proposed a way to dynamically adjust the threshold to be able to capture most of the particle peaks while still effectively removing background noise [6].

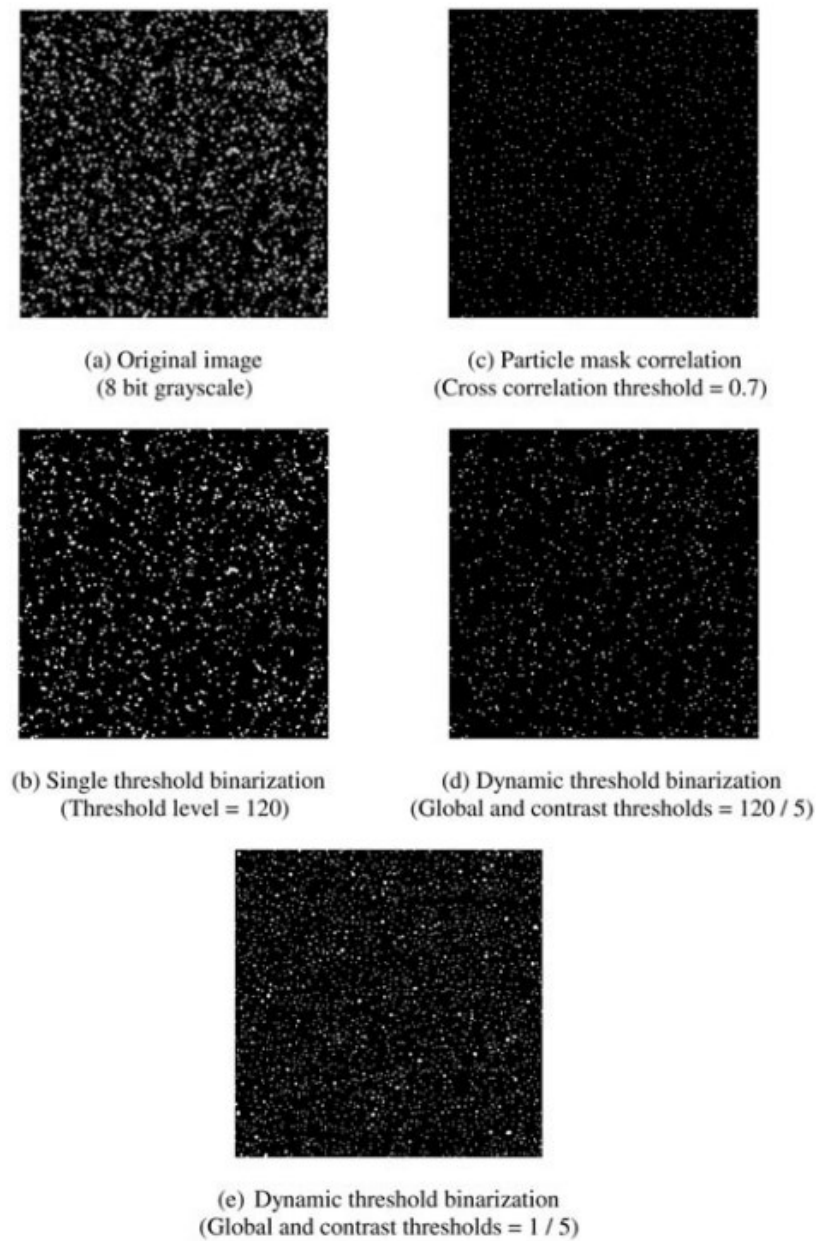


Figure 2.6 The comparison of Single threshold binarization and Dynamic threshold binarization [6]. It can be seen from (b) that using single thresholding would leave a large amount of particle out. In (e) for dynamic thresholding technique, a smaller number of particles are lost compared to (c)

2.5.2 U-Net Convolutional Network image segmentation

In recent years, various state-of-the-art computer vision techniques are based on deep convolutional neural network [14]. The use of deep convolutional neural network in this field has greatly outperformed traditional methods in image segmentation performance, especially under varying noisy conditions. The working fundamentals of U-Net consists of 2 steps, the encoder step and decoder step. First in encoder step, the input image is compressed by gradually decreasing spatial size and increasing convolutional filter size, reducing spatial scale of input information. In the second stage, the image original size is restored by the up-sampling process. In terms of U-Net network structure shown in Figure 2.7, are multiple sets of encoding/decoding steps, a bottleneck step and multiple shortcut [15].

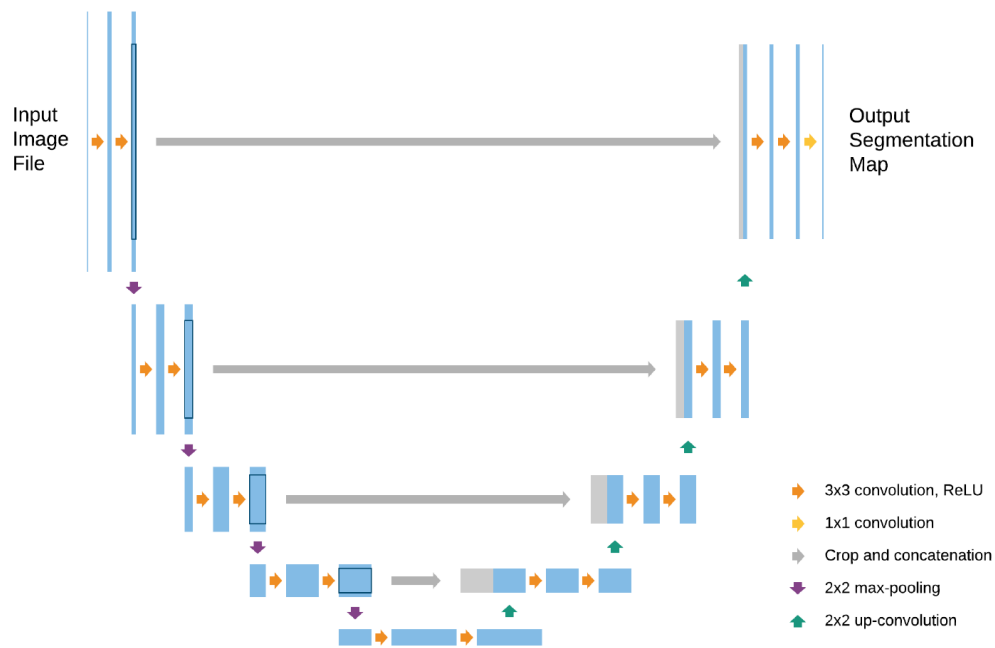


Figure 2.7 Basic U-Net architecture. The arrows represent the different operations, the blue boxes represent the feature map at each layer, and the gray boxes represent the cropped feature maps from the contracting path [16].

The U-Net image segmentation deep neural network, originally proposed by Ronneberger [17] is already implemented in MATLAB Computer vision toolbox through function *unetLayers* [18].

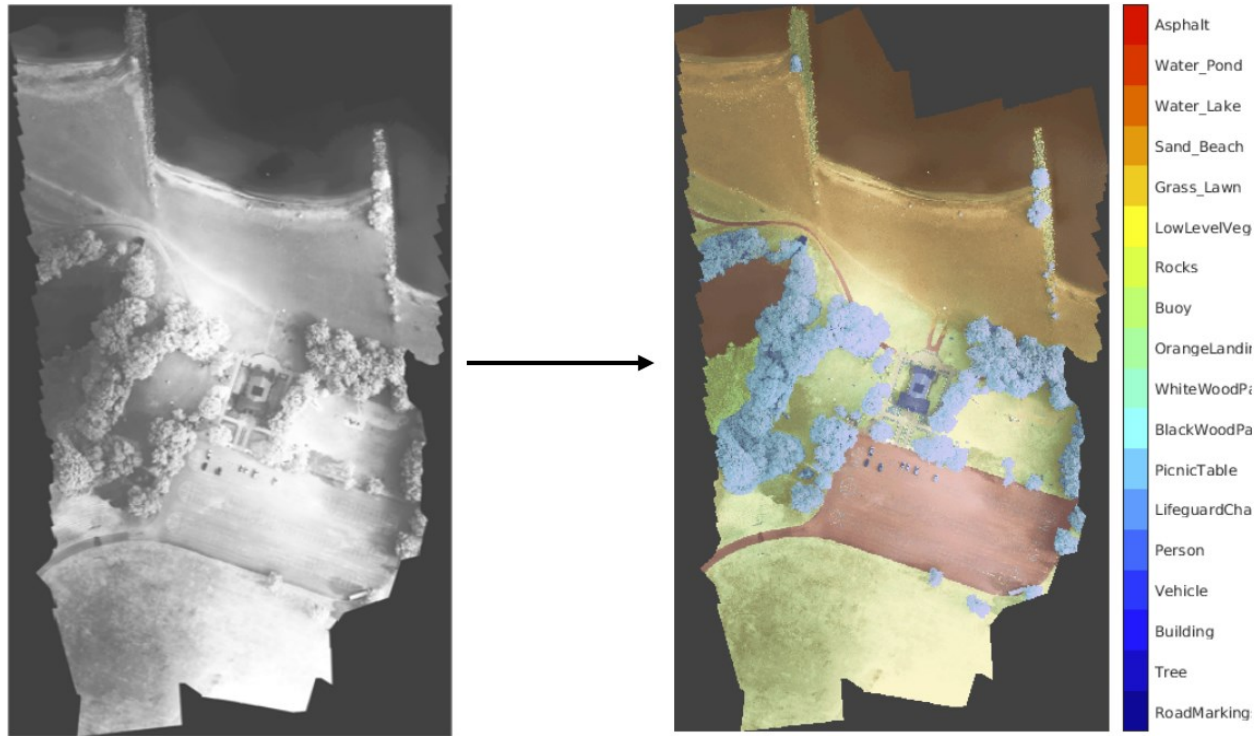


Figure 2.8 An example of Semantic Segmentation of Multispectral Images Using Deep Learning (U-Net) [19]. The trained U-Net algorithm label the multispectral image through recognition of spatially distributed structural information previously learned during the training.

2.6 GENERAL BACKGROUND ON IMAGE NOISE

The word "noise" refers to any undesirable and/or random occurrence that might damage an image by distorting its original information and change the outcome of the image from the original source.

In digital photography, it is an unavoidable fact that some kind of random variation in pixel value makes the captured image distorted from the actual image. This effect is known as “Image Noise” and this type of error cannot be eliminated due to current technology in digital camera. Image sensors, scanners, shot noise, and air turbulence are all typical causes of noise [20]. In digital camera, the noise can be categorized into 3 types, background noise, shot noise, and device noise [4]. These noises are considered random and not pixel independent, thus can be modelled using a simple random noise model described in Equation 2.2 [20].

$$I_n(i, j) = I(i, j) + n(i, j) \quad (2.2)$$

Where I_n represent noisy image that resulting from the addition of noise $n(i, j)$ on to clean image $I(i, j)$ [20].

Common noise model studied in PTV application are Poisson noise and Gaussian noise [9]. In previous studies on particle image with noise proposed by Lei et al., the Gaussian noise model is considered suitable in general PTV applications [9]. The paper suggested that whether the Poisson noise model or Gaussian noise model is used, the performance on particle localization accuracy of PTV using their software shows similar trends [9]. Therefore, the Gaussian noise model will be used in this research.

2.7 REMOVAL OF IMAGE NOISE

2.7.1 *Image Denoising Introduction*

In image processing, the image denoising technique has always been a challenging problem. The alteration of image data from various types of noises is undeniable during image acquisition, transmission, and storage. The image noise reduction technique or the action known as image denoising can be very important in removing these unwanted signal before the image can be used for further analysis [21].

In the most primitive way, a noise in natural image can be modelled as the Gaussian noise, whereas more complex data acquisition method such as MRI images and thermally generated electron sensors noise can be modelled using different distribution such as speckle noise and dark current noise [21]. Another type of common noise in digital camera imaging is shot noise, which is caused by the fluctuation of electrical current in the camera sensor described by Poisson statistical model [22] [23].

The main goal of the image denoising problem is to recover a clean picture x from a source that has been contaminated by some noise v . The noisy image y follows the degradation model $y = x + v$ where v is the random manifestation of noise in image describe in Equation 2.3:

$$y = x + v \quad (2.3)$$

The traditional method of removing noise includes non-local means filtering (NLM), block-matching and 3D filtering (BM3D) and Shearlets denoising [24]. These nonlinear noise filter aim to smoothen noisy components in the image while still retaining sharp edge in the images [25].

For instance, the BM3D technique denoise image by noisy patch estimation and calculate the weight contribution for each patch to overall image.

The key to accurately selecting denoising filters suitable for a particular application depends on the ability to recognize the statistical features of that specific type of noise.

Even though in the traditional denoising method, the adjustable parameters of various denoising filters can be adjusted such that they achieve a substantial boost in image quality that is subjected to various types of noise. It is essential that in each specific type of image denoising applications, a tremendous effort has to be done in order to fine-tuned suitable denoising filter parameters [25]. Lately, the expeditious development in Graphics processing unit (GPU), the progress in development of denoising technique based on deep learning framework has revolutionized the way scientist approach complex denoising task. The rapid growth in scheme of image processing using deep learning has ignite the other field such as image segmentation and object classification [23].

Traditional denoising method are considered content agnostic. This is because regardless of image structure the denoising behavior does not change. Denoising filters such as BM3D and nonlocal means exploit solely the statistical properties of noise in image to increase SNR of the denoised image [23]. One reason why nowadays the contemporary denoising method through the use of deep learning techniques are more successful is because the ability to fine-tune parameter adjustments to attain the feasible performance using known-good, noise-free image to train the denoising model to allow revolutionary transformation from noisy image to clean image. The denoising techniques involving known image structure are called content-aware denoising [23].

Still, the content-aware curtail the ability for an individual denoising filters to works on multiple types of image content [23].

Nevertheless, many modern denoising techniques do provide a step-up in digital image denoising performance in various applications, the investigation of the performance of modern image denoising using deep learning technique in particle tracking velocimetry application, to the best of my knowledge, is not precisely evaluated.

In the particle tracking velocimetry application, not only the perceptual quality of denoised image is critical, but also their ability to retain truthful information in the image without alteration in final outcome in particle localizing and particle tracking application.

2.7.2 Deep Convolutional Neural Network for Image Denoising (DnCNN)

One of the breakthrough in modern denoising algorithm is Deep Convolutional Neural Network (DnCNN) [26]. The most primitive use of deep neural network for image denoising using unsupervised learning procedure was proposed by *Seung and Jain*, where the proposed convolutional network can denoise image without the knowledge of noise level [27]. The use of deep learning, unsupervised approach can be advantageous compared to traditional denoising filter because it eliminate manual denoising parameters optimization to achieve good performance [26]. The deep convolutional neural network proposed by *Zhang et al.* (Figure 2.9) used the stacking multiple sets of convolutional layers, batch normalization layers and ReLU activation function. The stacking layers of Convolutional Neural Network (CNN) can increase the complexity of the denoising model, hence, the ability of exploiting accurate noise characteristics [26]. The proposed DnCNN model is mapping the noisy input image y to residual image v from noisy image (Equation 2.4).

$$y = x + v \quad (2.4)$$

Accordingly, the learned mapping function (Equation 2.5)

$$\mathbb{F}(y) = x \quad (2.5)$$

can be used to restore clean image when the input (y) is distorted by image noise [26].

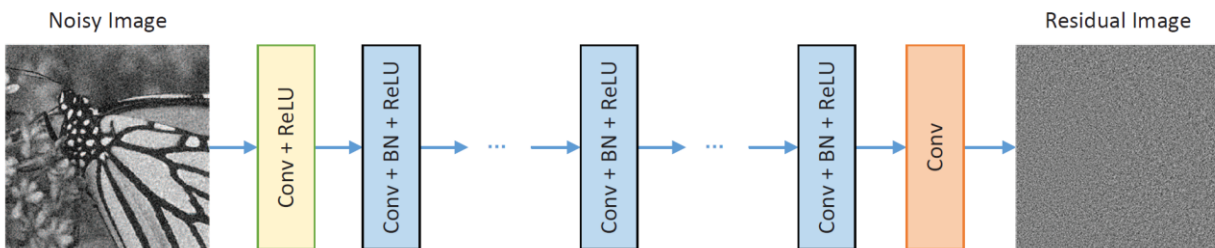


Figure 2.9 The architecture of DnCNN denoising network [26].

2.7.3 *Residual Dense U-Net Network for denoising (RDUNet)*

The U-Net model first proposed for image segmentation task despite the presence of noise, considering its exceptional performance in comparison to traditional image segmentation method [14]. Based on current state of neural network autoencoders, the U-Net can be considered the most wide-ranging in image restoration applications [28].

The development of Residual Dense U-Net Neural Network (RDUNet) for image denoising application leverage the unique trait of densely connected convolutional layers to reuse the feature maps is illustrated in Figure 2.10 and Figure 2.11 [28]. The contraction and expansion of spatially distributed information through encoding layers and decoding layers can be also modified to predict clean image from noisy image. The U-Net contraction path of the model consists of several convolutional layer which is followed by max-pooling layers, this operation down sampled the feature maps by half the original size, however, the number of feature map is doubled in order to prevent the loss of information [28]. In the middle of the network, it consists of bottleneck level where it served as a nonlinear dimensionality reduction of input feature. From bottleneck layer, it is connected to the decoding layer, the decoding layers perform up sampling of input data through the transpose convolution process [28]. The precise configuration of the network can be found on the original paper [28].

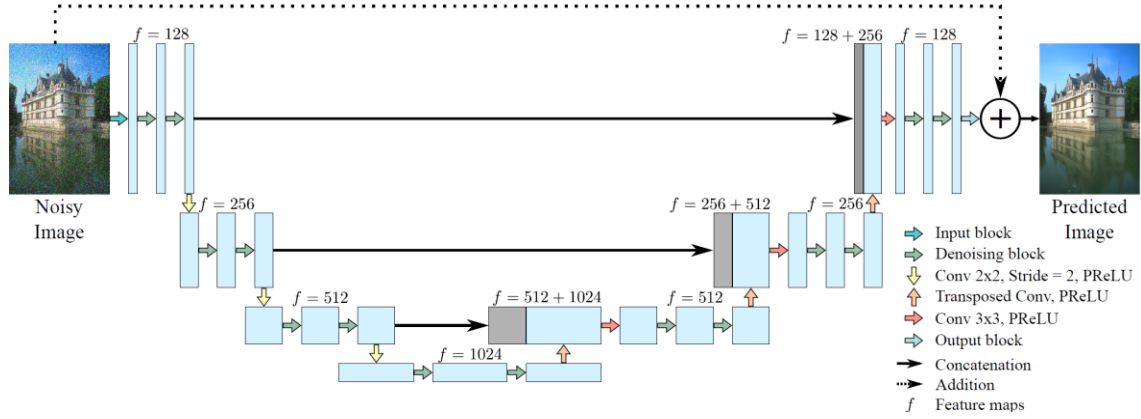


Figure 2.10 Architecture of RDUNet Neural Network [28].

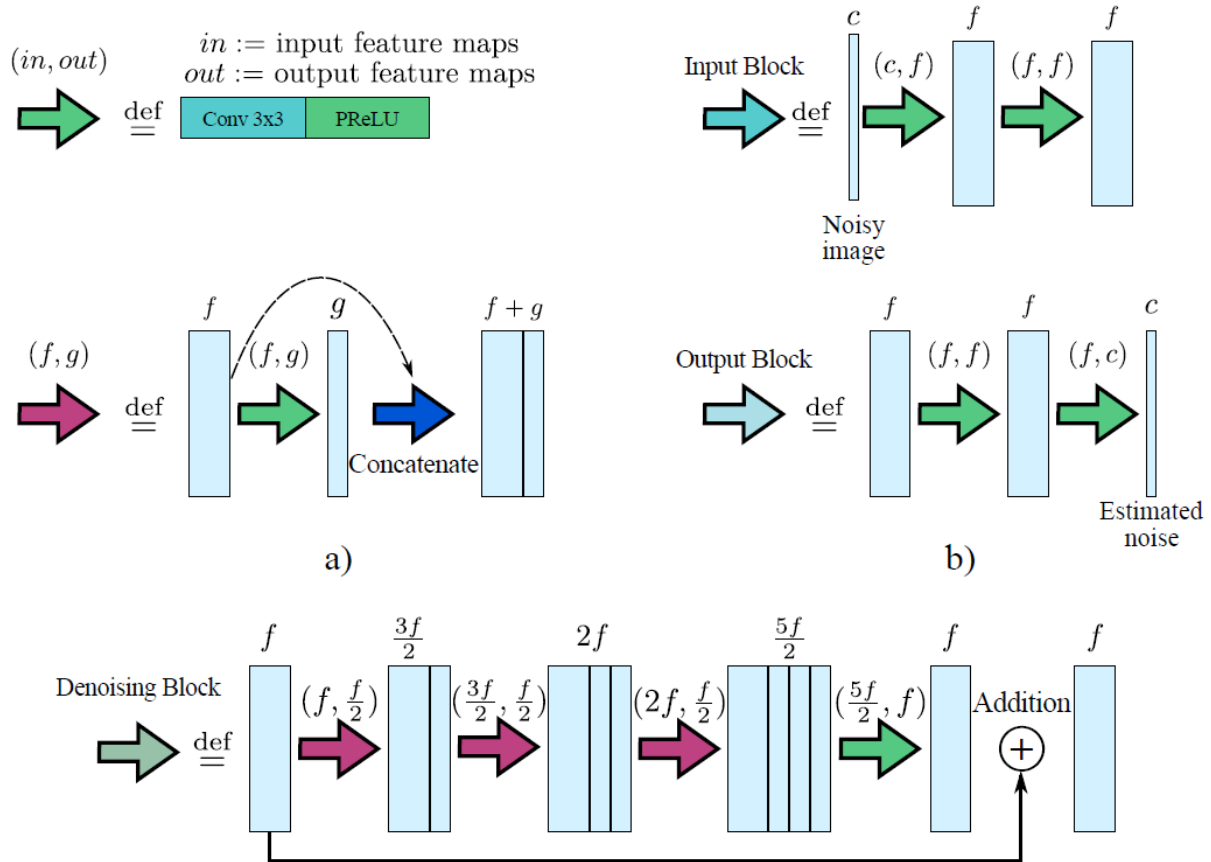


Figure 2.11 Building blocks of the RDUNet: a) Basic components used in the building blocks of the RDUNet model, b) input and output blocks, c) densely connected denoising block used throughout the model [28].

For the application of additive white Gaussian noise (AWGN), the advantage for this method proposed by [28] does not require any prior information about the level of the noise. The strong advantage of using convolutional layer in both encoding path and decoding path is to encode the spatial knowledge of the image noise distribution. According to the paper, this enables the model to be able to predict each individual pixel value for the exact noise value [28].

In comparison to DnCNN denoising model benchmark, the Residual Dense U-Net (RDUNet) clearly present progression in the denoised image quality. The paper concluded that in contrast to DnCNN, RDUNet does not perform any sub-sampling during the denoising process, so the spatial dimension is constant throughout, resulting in spectacular denoising performance [28].

2.7.4 Comparison of modern denoising technique

In order to select the most probable denoising technique for PTV applications, we reference the study conducted by *Gurrola-Ramos et al.* that ranked the performance of each denoising technique based on Peak Signal to Noise Ratio (PSNR) [28]. The PSNR metric is a commonly used parameters in quantifying the quality of reconstructed image that was corrupted by noise.

Methods	BSD68					
	$\sigma = 10$		$\sigma = 30$		$\sigma = 50$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BM3D	33.32	0.9158	27.75	0.7731	25.60	0.6858
U-Net	33.47	0.9224	28.16	0.7948	26.04	0.7152
DnCNN	33.88	0.9270	28.36	0.7999	26.23	0.7189
IRCNN	33.74	0.9262	28.26	0.7989	26.19	0.7171
FFDNet	33.76	0.9266	28.39	0.8032	26.29	0.7245
RDN	34.00	N/A	28.56	N/A	26.41	N/A
DHDN	33.42	0.9213	28.55	0.8110	26.44	0.7296
DIDN	33.98	0.9284	28.58	0.8075	26.47	0.7310
RDUNet	33.98	0.9297	28.59	0.8104	26.48	0.7347
U-Net+	33.57	0.9236	28.22	0.7966	26.10	0.7176
RDN+	34.01	N/A	28.58	N/A	26.43	N/A
DHDN+	33.50	0.9230	28.59	0.8120	26.47	0.7308
DIDN+	34.01	0.9286	28.61	0.8081	26.50	0.7320
RDUNet+	34.01	0.9300	28.62	0.8112	26.51	0.7358

Figure 2.12 Performance of different denoising methods on BSD68, Kodak24 and Set12 grayscale datasets with different noise levels [28]

According to the denoising performance table proposed by *Gurrola-Ramos et al.* in Figure 2.12, the PSNR score clearly demonstrate the superior removal of noise performance using deep-learning approach in image compared to traditional approach such as the BM3D [28].

In this thesis, we chose the best two performing deep neural network denoising algorithm for performance evaluation, the DnCNN and RDUNet. In contrast to commonly used performance matrix PSNR in image denoising, we chose to use particle location error to evaluate which of the denoising algorithm can provide the least amount of location error.

Chapter 3. METHODOLOGY

This chapter propose developed technique to overcome both the location inaccuracies and the loss of particle when the particle image is corrupted by image noise. Several methodologies are proposed for overlapping particle image (Point Spread Function Model Inversion) and non-overlapping particle image (Iterative Particle Reconstruction) at constant particle radius. Also, an attempt to identify particles that have random size is also presented in this chapter.

3.1 PROPOSED PARTICLE IMAGE IDENTIFICATION ALGORITHM

The software was developed on MATLAB environment to extract particle coordinate from particle image in presence of synthetic additive white Gaussian noise. Because of memory constraint, the image is broken down into multiple image patch of size 32 by 32 pixels. Then, a modified U-Net, which is a deep convolutional neural network image segmentation algorithm, is used to label each pixel whether they are particle or background. In the next step, a RDUNet denoising network is used to remove noise from image. Once the image is denoised, a particle image reconstruction using point spread function (PSF) further identify the particle peaks. For noisy particle images, it is recognized that the detected particle peaks contain a large number of false detection due to variation pixel intensity that was caused by noise. In order to remove these false detections, a distance transform is performed on particle-background binary label to calculate the likelihood of a particular particle being a true detection. Any detection that under the pre specified threshold are removed. Following that, the duplicate detections are removed using DBSCAN. Finally, the processed particle image sections are merged together to produce final results. The approach is shown schematically in Figure 3.1.

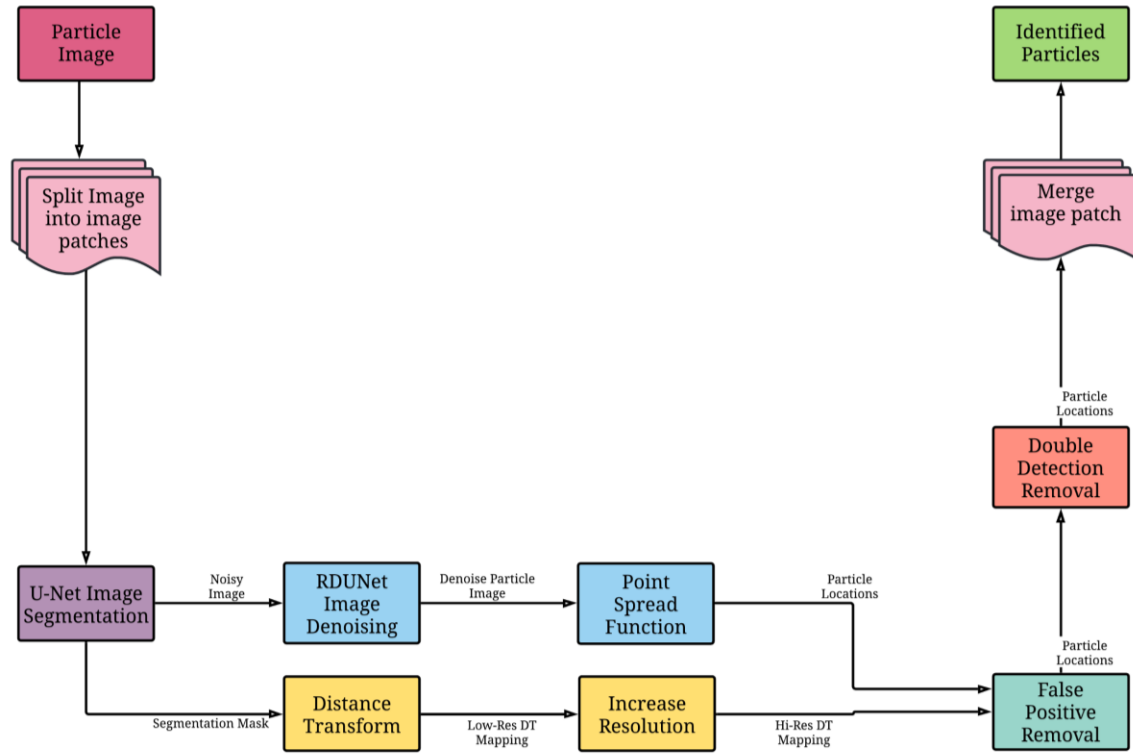


Figure 3.1 Process flow of our proposed particle image identification algorithm.

3.1.1 *Split Image into 32 by 32 pixel*

At the current level of in computer technologies, the particle image cannot be processed directly as a single image. This is because of the system memory used to perform particle location recovery +using point spread function (PSF) model. Therefore, the particle image has to be segmented into a stack of smaller image patch of size 32 by 32 pixels. The original image is cropped into image patches and saved on a system drive for later use in the proposed workflow.

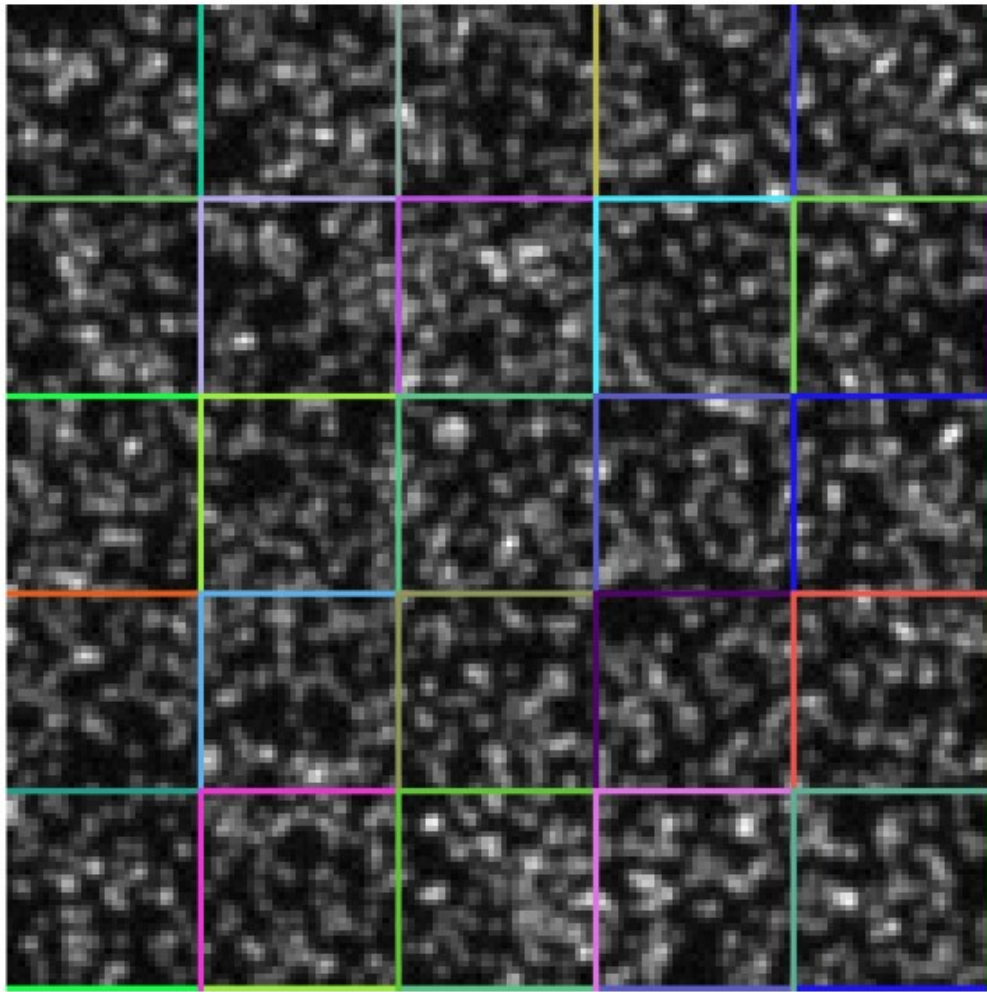


Figure 3.2 The original image of size 160 by 160 is spitted into smaller image patch of size 32 by 32 and individually processed through the proposed workflow.

3.1.2 Particle Image Segmentation using U-Net

After the original particle image is acquired, it is split into image patches of size 32 by 32 pixels. The U-Net deep neural network is utilized for labeling pixel in the image whether it is a particle or background. The U-Net is generated using MATLAB Deep Learning Toolbox built in function *unetLayers* [18]. The depth parameters of U-Net describe the number of times the downsamples/upsamples process on the input image [18]. The depth of 3 is chosen for this image segmentation task. This depth value is suggested by MATLAB *unetLayers* function documentation [18]. The plot of actual U-Net model is shown in Figure 3.3 and the parameters that used to create the model is described in Table 3.1.

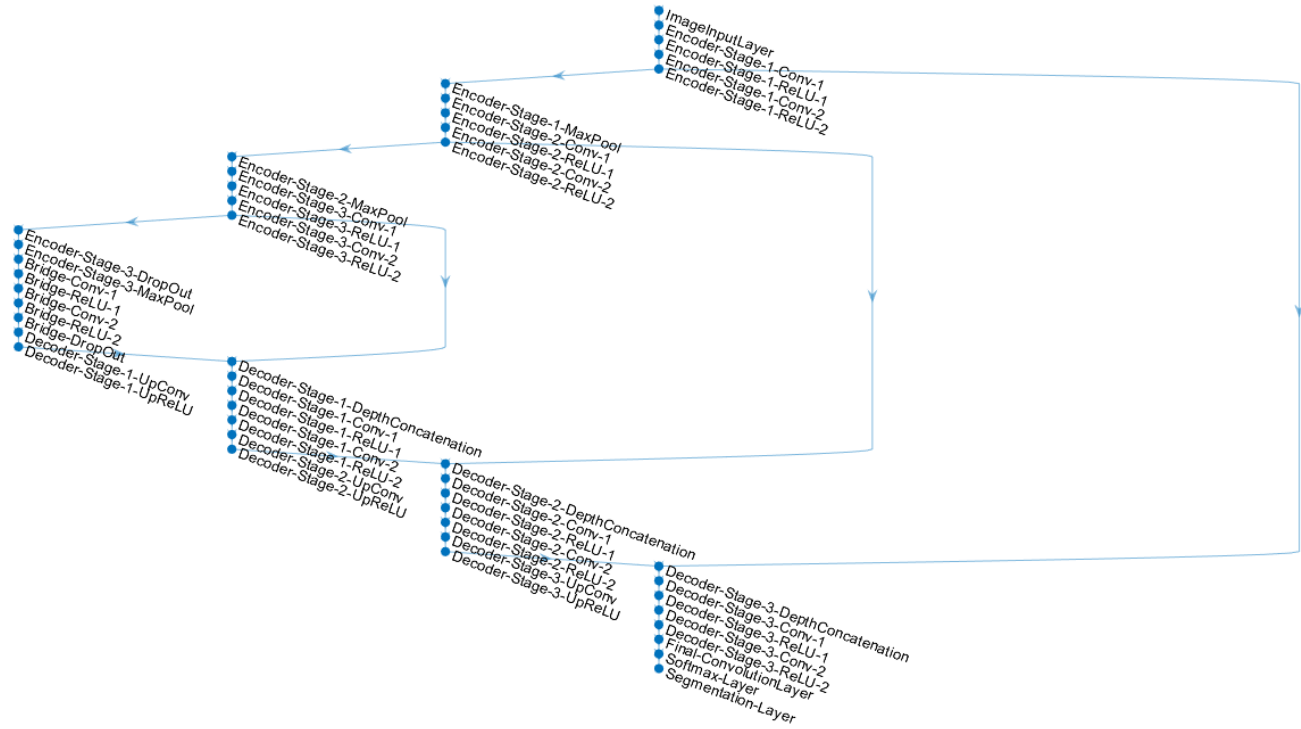


Figure 3.3 The plot representation of U-Net model for particle image segmentation task.

Table 3.1. Critical Parameters for U-Net model creation

Parameter	Value	Unit
Number of Class	2	images
Encoder Depth	3	pixel

Table 3.2. Image Patch Generation for U-Net training

Parameter	Value	Unit
Number of Noisy-Label image pairs	100,000	images
Patch size	32 by 32	Pixel
Noise percentage	Random 1 to 6	Percent
N_{ppp}	Random 0.01 to 0.10	-
σ	1	Pixel

Remark: N_{ppp} refers to the particle density per unit pixel. (Particle Per Pixel)

For the U-Net training process, the algorithm only receives Noisy-Label image pair shown in the Figure 3.4. Also, the algorithm does not have access to the random noise level corresponding to the training image pair. The denoising model is trained on MATLAB platform using default hyperparameters suggested by MATLAB Image Segmentation Online example [29].

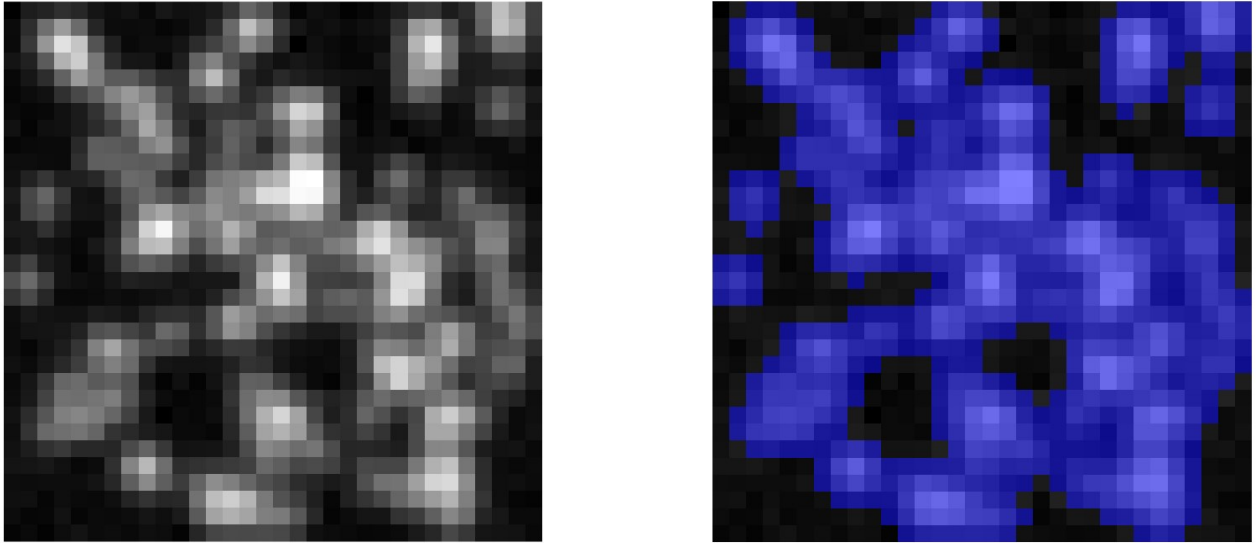


Figure 3.4 Randomly selected Noisy-Label training image pair used for U-Net algorithm training. Left: Noisy image as input for U-Net. Right: Binary label (overlayed on noisy image for illustration purpose only).

The training Root Mean Square Error (RMSE) and loss values are manually monitored until they stop decreasing then stop training process. The operating system of the workstation used is Windows 10 64-bit with AMD Ryzen 9 3950X CPU and 32 GB of DDR4 system memory. The training time was approximately 2 hours in a Nvidia RTX 3060 GPU. During training, it is observed that the training loss and validation loss are similar in magnitude, this means that the model does not overfit the training data. The training hyperparameters are described in the Table 3.3.

Table 3.3. U-Net Image Segmentation Training Parameters

Training Parameters	Value
options.Momentum	0.9
options.InitialLearnRate	1.00E-03
options.LearnRateSchedule	'none'
options.L2Regularization	1.00E-04
options.GradientThresholdMethod	'l2norm'
options.GradientThreshold	Inf
options.MaxEpochs	30
options.MiniBatchSize	128
options.Verbose	1
options.VerboseFrequency	2
options.Shuffle	'once'
options.ExecutionEnvironment	'cpu'
options.Plots	'none'
options.SequenceLength	'longest'
options.SequencePaddingValue	0
options.SequencePaddingDirection	'right'
options.DispatchInBackground	0
options.ResetInputNormalization	1
options.BatchNormalizationStatistics	'population'

3.1.3 Particle image denoising using Residual Dense Neural Network (RDUNet)

The next step is to denoise the image patch. In this thesis, we compare RDUNet and DnCNN denoising algorithms and pick the best performing algorithm to use in the image denoising task. According to the paper published by *Gurrola-Ramos et al.*, RDUNet clearly shows superior denoising performance in comparison to DnCNN on human image [28]. However, there are a non-negligible possibility that in this particular image structure, DnCNN could outperform RDUNet in terms of final particle location accuracy, so they both were chosen for comparison.

The RDUNet model is modified from the source code of the paper ‘A Residual Dense U-net Neural Network for Image Denoising’ published by *Gurrola-Ramos et al.* [28]. The model was originally coded on opensource PyTorch machine learning library and it is available on the internet [30]. In order to be compatible with the rest of the other particle identification steps described in the workflow, the model is converted from PyTorch platform to Open Neural Network Exchange (ONNX) platform [31]. After the model is exported, the model is imported to MATLAB through built-in function on MATLAB Deep Learning Toolbox `importONNXNetwork` [32]. The only change to the model is the size parameter of the input layer, it is modified to 32 by 32 by 1 in order to match the size of image patch throughout workflow.

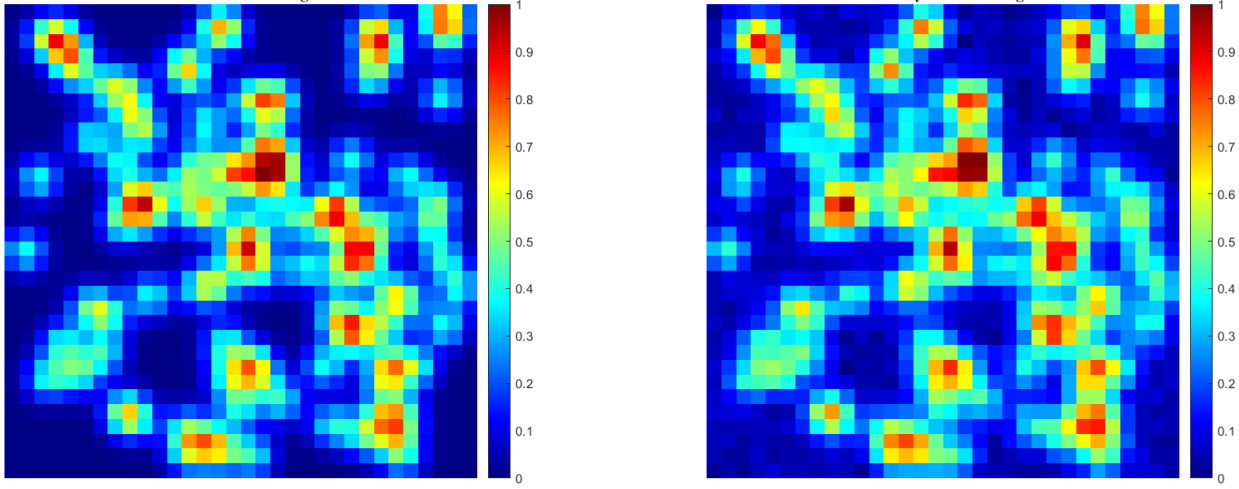


Figure 3.5 Randomly selected image pairs for training. In this image pair, the N_{ppp} 0.10 and 5% noise. Left: Clean particle image. Right: Noisy particle image with 5% of added Gaussian noise. Images are colorized based on their pixel value for illustration purpose.

For the training process of RDUNet, the clean and noisy particle image patches generated, as illustrated in Figure 3.5, are split into the ratio of 0.95 and 0.05 for training set and validation set. Since the goal of RDUNet is to map the noisy image y to the clean image x according to the Equation 2.1, the training process requires clean image patch and noisy image patch. The input is noisy image patch and output is clean image patch. The synthetic image patch generation parameters are described in the Table 3.4.

Table 3.4. Image Patch Generation for RDUNet training

Parameter	Value	Unit
Number of Noisy-Clean image pairs	100,000	images
Patch size	32 by 32	Pixel
Noise percentage	Random 1 to 6	Percent
N_{ppp}	Random 0.01 to 0.10	-
σ	1	Pixel

For the training process, the algorithm only receives Noisy-Clean image pair, the algorithm does not have access to the random noise level corresponding to training each image pair. The denoising model is trained on MATLAB platform using hyperparameters similar to the paper proposed by *Gurrola-Ramos et al* and they are shown in Table 3.5 [28].

Table 3.5. RDUNet Training Parameters

Training Parameters	Value
options.Momentum	0.9
options.InitialLearnRate	1.00E-05
options.LearnRateSchedule	'piecewise'
options.LearnRateDropFactor	0.2
options.LearnRateDropPeriod	27
options.L2Regularization	1.00E-05
options.GradientThresholdMethod	'l2norm'
options.GradientThreshold	Inf
options.MaxEpochs	27
options.MinibatchSize	8
options.Shuffle	Inf
options.SequenceLength	'longest'
options.SequencePaddingValue	0
options.SequencePaddingDirection	'right'
options.DispatchInBackground	0
options.ResetInputNormalization	1
options.BatchNormalizationStatistics	'population'

The training Root Mean Square Error (RMSE) and loss values are manually monitored until they stop decreasing then stop training process. The operating system of the workstation used is Windows 10 64-bit with AMD Ryzen 9 3950X CPU and 32 GB of DDR4 system memory. The training time was approximately 48 hours in a Nvidia RTX 3060 GPU. During training, it is observed that the training loss and validation loss are similar in magnitude, this means that the model does not overfit the training data.

3.1.4 *Particle image reconstruction from particle image through model inversion*

The implementation of Equation 2.6 can be done with ease using MATLAB internal function *lsqnonneg*. With reference to Cherminet et al. studies mentioned in Chapter 2, the selected value of 20 for Δ_{E^-} is used throughout the proposed particle identification workflow. Since the inversion matrix can be pre-calculated for a given value of σ , the weight matrix is calculated and saved on disk for faster processing time in each image patch. The Figure 3.6 illustrates the visual representation of the particle image and its associated light point source representation.

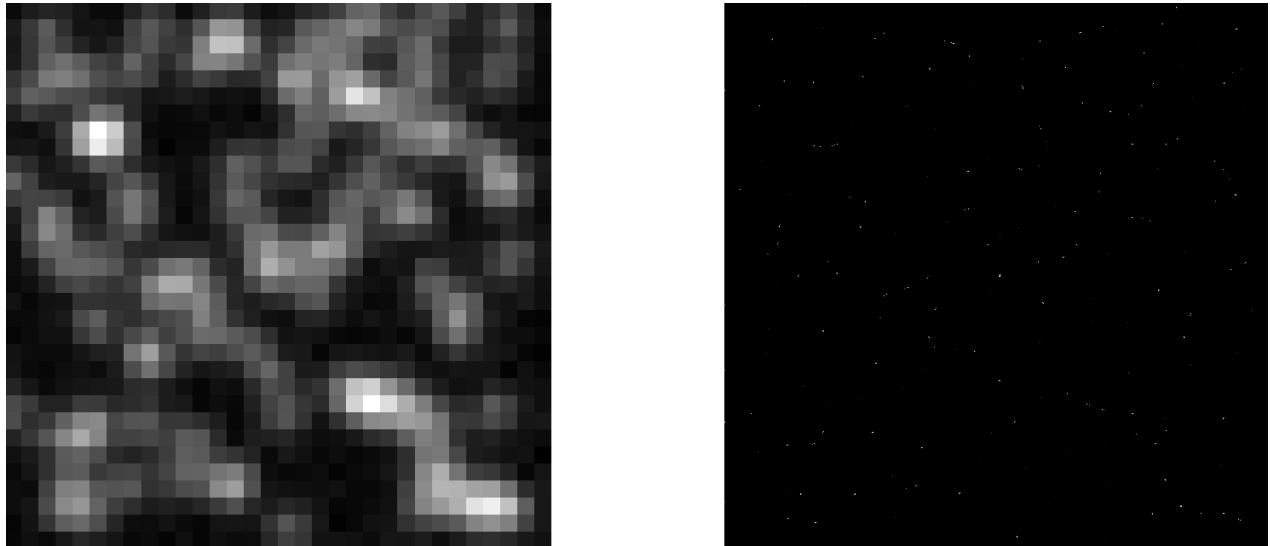


Figure 3.6 The sparse (right) particle representation of the particle image (left).

3.1.5 Particle image segmentation using U-Net.

The contribution of noise in the real image lowered quality of information that we can extract from particle image. Image noise can significantly increase the number of False Positives where there is no particle present, or multiple detection of same particle. With recent development in computer vision task, not only can we classify images from one another, but we can also use the deep neural network to enable us to accurately localize specific feature on image to yield very precise segmentation map. By using image segmentation technique, the False Positive detection due to the addition of noise can be removed (Figure 3.7).

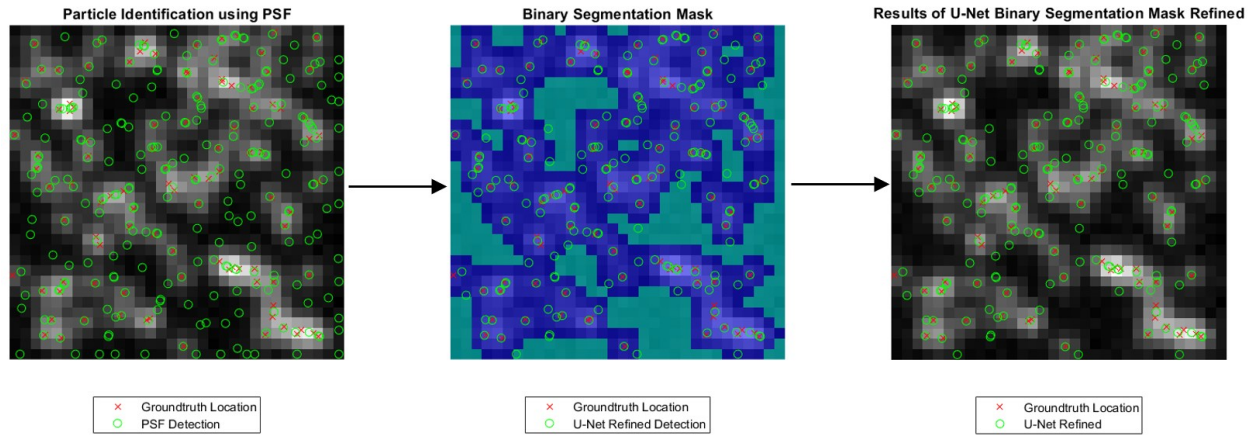


Figure 3.7 Visual illustration of background False Positive (FP) particle rejection. Left: Unprocessed detections from PSF. Mid: U-Net binary mask indicate the region where particles are present. Right: Output of the algorithm.

The state-of-the-art image segmentation technique was proposed by *Ronneberger et al.*, originally for biomedical image segmentation application [17]. The proposed network relies on contraction-expansion path of convolutional neural network to first condense only the critical feature of the image into vectors, and later reconstruct these vectors back into the classified regions of the input image.

3.1.6 *Reliability improvement using distance transform.*

Given the information obtained from the image segmentation step, we can produce a binary mask whether a given pixel is a background pixel or particle pixel. By using the binary mask, we proposed an additional method to further remove False Positives (FP) among all detection. This removal of False Positives is based on the assumptions that the particle peak is located at the greatest distance from the edge of the binary mask. With the same reasoning, any particle that was located in close proximity to the edge of the binary mask had a higher possibility that it would be a False Positive (FP) detection. In order to increase yield and reliability performance metrics, the score associated to each detection is calculated based on their distance from the edge of the binary mask.

The paper was proposed by *C. R. Maurer et al.* and suggested the method to compute the exact Euclidean distance transforms the binary images [33]. The algorithm of calculating distance transform binary image is directly implemented in MATLAB as the function *bwdist* [34].

Since the size of particle image is in the order of several pixels, it is relatively small in terms of pixel distance. Therefore, the calculated distance transform mask is scaled up by 20 times the original size through bicubic interpolation. The bicubic interpolation takes the average of its 4 by 4 nearest neighborhood. As a result, the upscaling effect of binary mask, as illustrated in Figure 3.8, enables the possibility of subpixel resolution value to accurately reject False Positive that was located in close proximity to the edge of the binary mask with greater precision.

It is a critical task to determine the exact distance threshold from edge which effectively removes False Positives while maintaining maximum possible yield without sacrificing the reliability. Therefore, we explored the range of distance threshold by means of yield and reliability to determine the best tradeoff between False Positive detection and False Negative (FN) detection. In this thesis, the symbol γ_{dist} is denoted for the score associated to each detection, and γ_{min} is denoted for minimum threshold that it would consider true detection. The value γ_{optim} has to be determined experimentally.

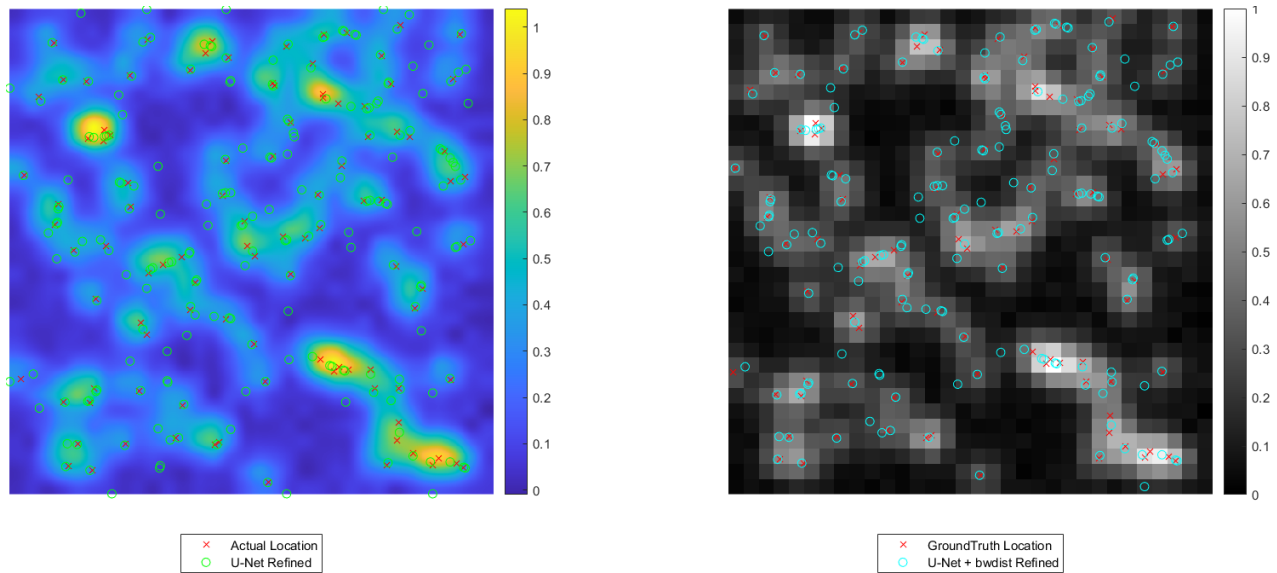


Figure 3.8 Left: The distance transform of binary mask from U-Net is scaled up 20 times to increase resolution. Right: The clean-up detection by rejecting any detection that below the γ_{optim} threshold.

3.1.7 Reliability improvement using density based spatial clustering.

To further remove False Positives, caused by noise in image, we propose the idea of removing double-detection of particle using density based spatial clustering (DBSCAN). DBSCAN is a very versatile tool in MATLAB to perform data clustering using density-based criteria. The fundamental working of DBSCAN is that for a given sets of points in space, it calculates the distance to its neighbor within a specific radius and group closely packed points into a cluster. Where points in space that are not in close proximity to other points in space, the DBSCAN algorithm classify as an outlier [35]. In addition to the density of point in a given space, the DBSCAN also consider the minimum number of points in the neighborhood to be over a specified threshold. An example of DBSCAN clustering is presented in Figure 3.9.

By leveraging the clustering algorithm, we can utilize DBSCAN to identify the detections that are too closed together and combine into a single particle. A centroid of set of points that were identified in a cluster are calculated.

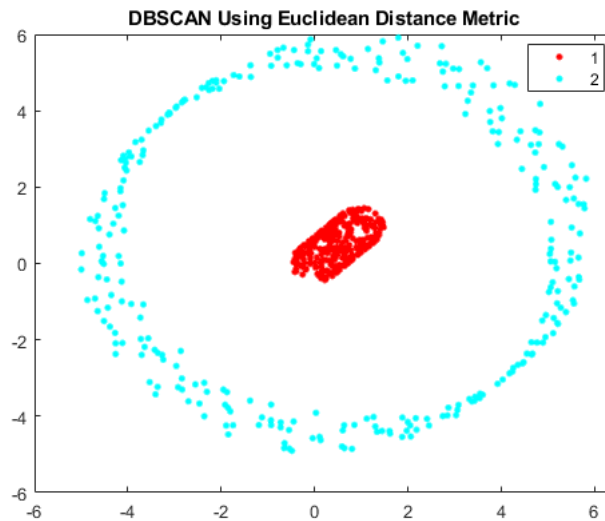


Figure 3.9 Example of clustering of points using DBSCAN algorithm [36],

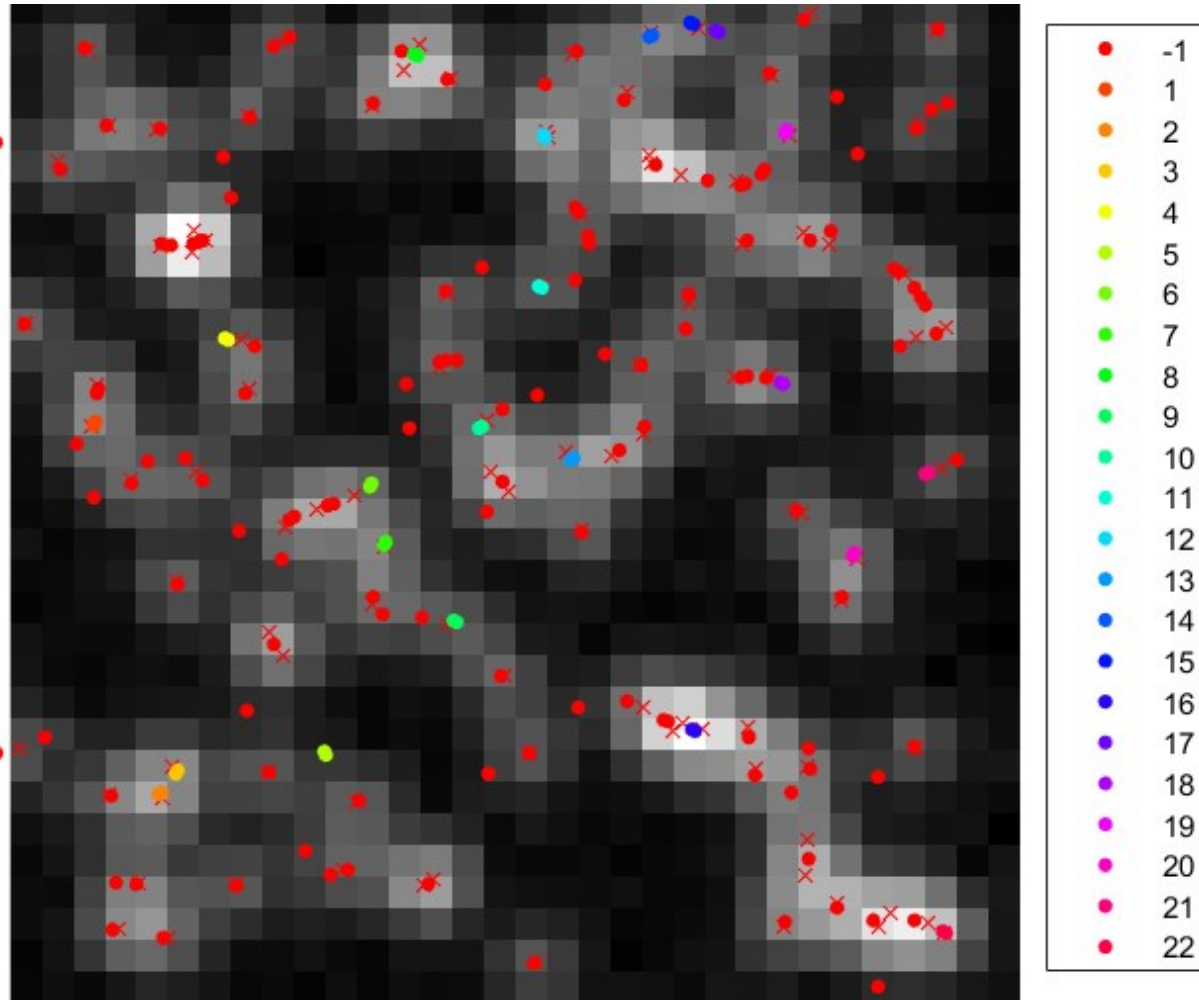


Figure 3.10 The detections of particle in noisy image through PSF method that already refined by U-Net and distance transform technique. Each number corresponding to specific color represent the clustering of points using DBSCAN algorithm. The number less than 0 indicate outliers, whereas the number greater than 0 indicate multiple detection of the same particle. Hence the centroid of each cluster is calculated. The red cross indicates the ground truth position of each particle.

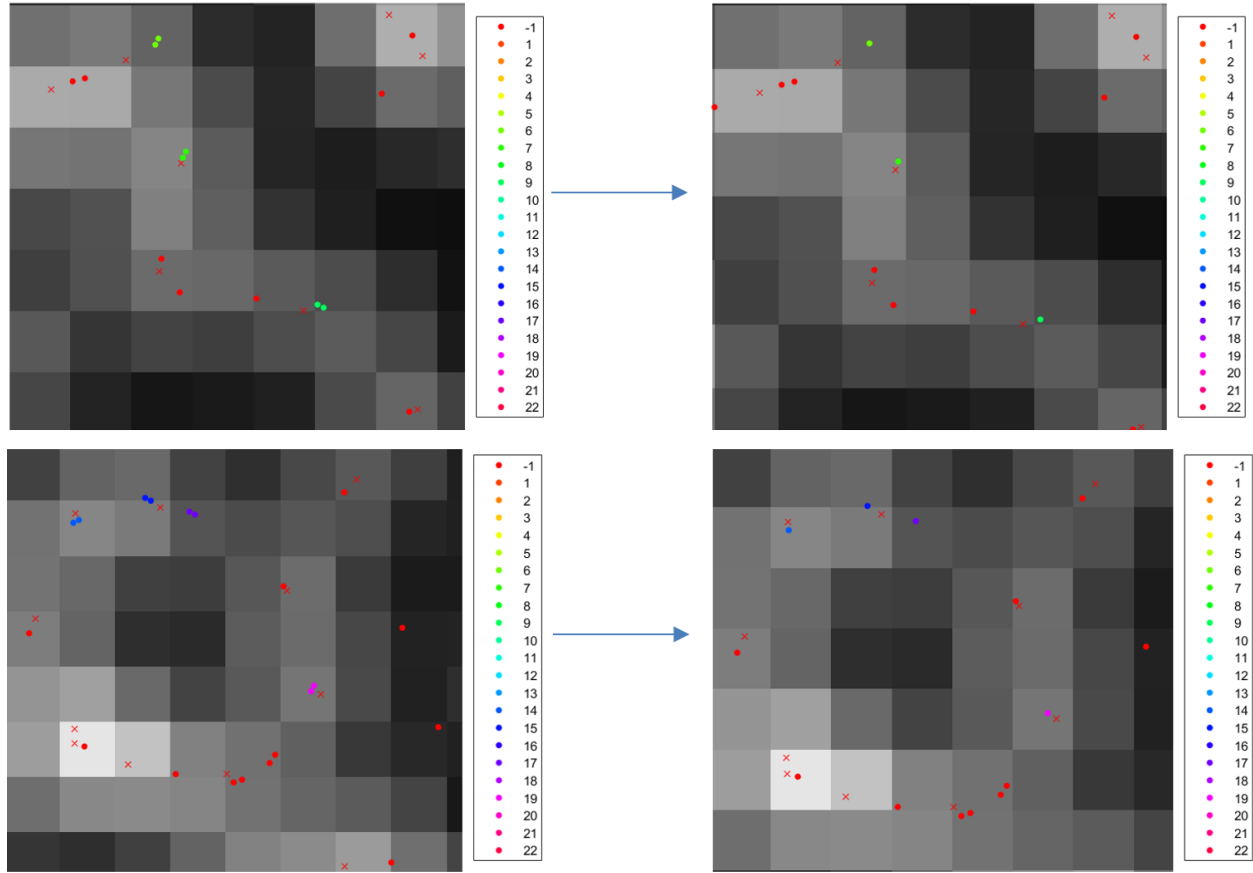


Figure 3.11 Top: Zoom-in version of a particle image. The centroid of cluster 5, 6, and 7 are computed. Bottom: The centroid of cluster 14, 15, 18 19 are computed.

In order to determine the proper distance threshold for DBSCAN to consider as a single cluster, the range of distance threshold is performed on synthetic image. The distance threshold denoted as γ_{dbscan} of DBSCAN algorithm is optimized in order to get the optimum point in terms of yield and reliability.

3.2 SYNTHETIC IMAGE GENERATION

In order to assess the performance of particle identification, the synthetic particle images are used. This is because it is not feasible to know the exact particle location in experimental image. The synthetic images are synthesized to closely resemble the representative experimental condition. The synthetic image was generated by randomly placing a Gaussian particle, using superposition principle, with particle radius σ which is described in Equation 3.6 on a 2-dimensional space of size 32 by 32 pixels until the desire particle density N_{ppp} is reached. Each particle has the same intensity with the maximum intensity (I_{max}) of 1.

$$I(x, y) = I_{max} \exp \left(- \left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2} \right) \right) \quad (3.6)$$

After the desirable particle density N_{ppp} is reached, the image intensity is rescaled such that the range of pixel value is from 0 to 1 using MATLAB function *rescale*.

In order to assess the robustness of our proposed particle identification algorithm under range of seeding density and noise levels, the synthetic image is subjected to Additive White Gaussian Noise (AWGN) using MATLAB function *imnoise*. The variance of Gaussian noise is from 1 to 5 percent with zero mean. The noise level percentage is defined such that the standard variation of added noise σ_{noise} is a percentage of the total dynamic range of 0 to 1. For performance testing on larger image size, the synthetic images are generated using the same procedure as the 32 by 32 pixels image patch, but the size was increased to 160 by 160 pixels. There are 10 images patches for each specific configuration tested for 32 by 32 pixel. Due to a longer computational time for larger image size of 160 by 160 pixels, the number of samples per configuration is 2 samples.

3.3 PARTICLE IDENTIFICATION PERFORMANCE METRICS

We chose to use commonly used performance metrics to evaluate the particle identification and localization performance, suggested by *Ruhnau et al.* [37]. The performance parameters are the yield, reliability, and localization error. The criterion proposed by *Ruhnau et al.* to be considered a True Positive, each detection has to satisfied 2 criterions.

1. The detection is within the specified distance from the ground truth location.
2. There is no particle closer than itself to the ground truth location.

The distance tolerance for satisfying condition 1 is set to be 0.5 pixel [37]. For the correctly identified particle, it is considered a True Positive (TP). For any identified particle that do not meet the criteria above, it is considered False Positive (FP). In the case that a particle is not detected in the neighborhood, it is considered False Negative (FN). The yield parameter is the ratio between the correctly identified particle (TP) divided by the total number of particles. The reliability parameter is defined to be the ratio between the correctly identified particle (TP) divided by the total number of detections [37] [38].

$$Yield = \frac{\#TP}{\#TP + \#FN} \quad (3.7)$$

$$Reliability = \frac{\#TP}{\#TP + \#FP} \quad (3.8)$$

In terms of localization accuracy of particle, the mean localization error, which can be calculated from the sum of Euclidian distance between particle detected and ground truth location divided with total number of True Positive (TP) [38].

$$Mean\ Localization\ Error = \frac{1}{\#TP} \sum_{i=1}^{N_{TP}} \|x_{TP_i} - x_{P_i}\| \quad (3.9)$$

3.4 RESIDUAL NOISE PROFILE OPTIMIZATION USING ITERATIVE PARTICLE RECONSTRUCTION TECHNIQUE

In order to further improve the location accuracy, we proposed another method to extract particle location. We perform reconstruction of particle and iteratively changing particle location and calculate the difference between generated image and original image. Theoretically, the only difference between the two should be only the noise. Nonetheless, this method cannot process the particle image effectively in overlapping particles, so the results are excluded from the method proposed in Section 3.1. The diagram of this method is presented in Figure 3.12.

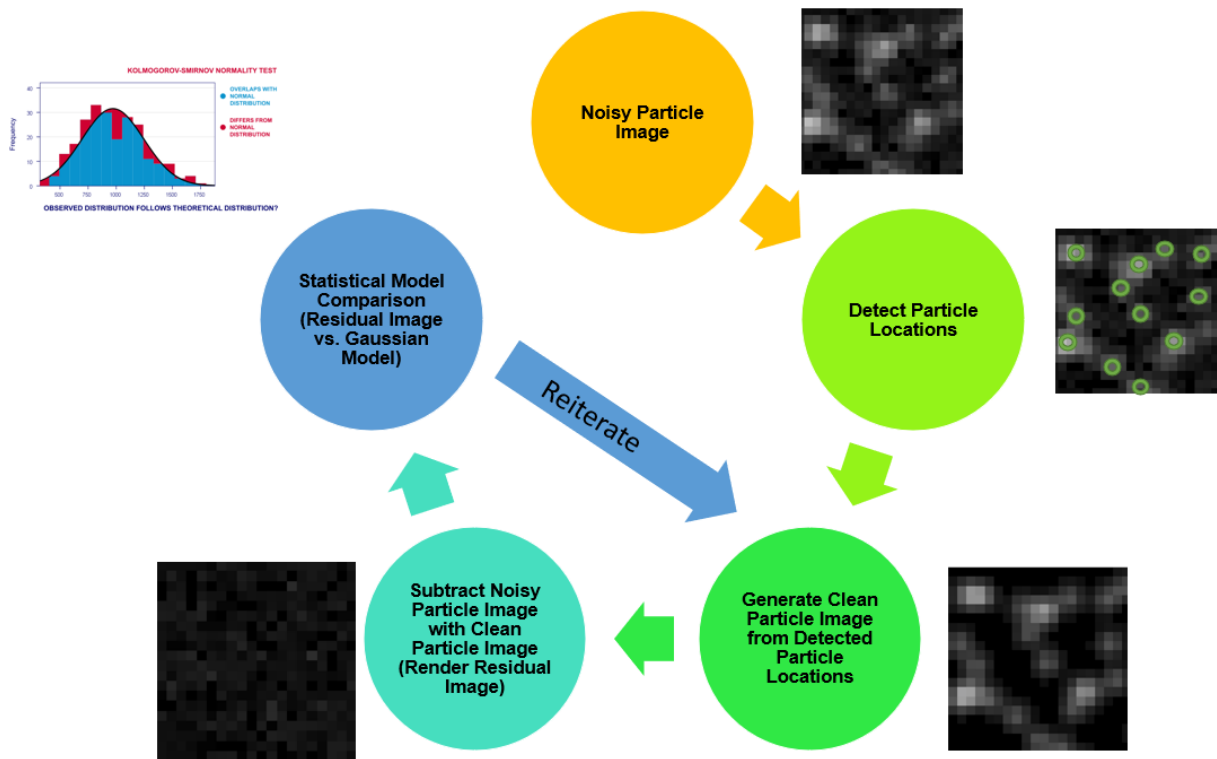


Figure 3.12 Proposed residual noise optimization workflow for non-overlapping particle image.

3.4.1 *Reconstruction procedure*

First, U-Net image segmentation is used to locate the particle location. This would result in a blob of connected pixels containing a particle (Figure 3.13). The centroid of each blob is calculated to obtain approximate coordinate for each particle location. Then, using the initial estimated coordinate, the least square fit of Gaussian intensity is performed on particle. Next, the Gaussian residual optimization algorithm iteratively reconstruct noise-free particle image. The reconstruct image is subtract from noisy image, resulted in only the noisy component of the image. The residual is then compared with Gaussian noise model by performing K-S test. We then use optimization algorithm to iteratively reconstruct particle image location to determine the best coordinate possible to have residual image fit to Gaussian noise model as much as possible. Once the tolerance criteria are met, particle coordinate from optimization are considered error corrected.

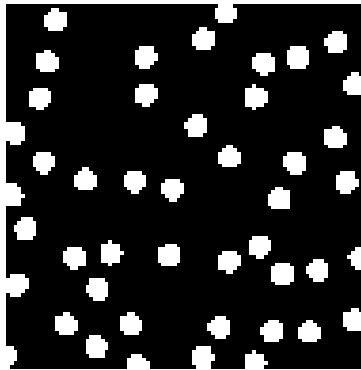


Figure 3.13 The binary blob containing particle. Every single pixel in each blob is used to calculate a centroid corresponding to a particle.

Based on the assumptions that particle image follows Gaussian intensity distribution, and they are uniform in size and shape, the particle coordinates estimated in previous step are used to recreate artificial particle image which resemble noisy image. By performing image subtraction between

noisy image and artificial clean image, the noisy component can be extracted. However, if the initial coordinate estimated from previous step is not accurate, the resulting residual image will not have Gaussian property. Therefore, by leveraging on degree of Gaussian in residual image, the particles coordinates can be iteratively adjusted to recreate artificial particle image such that the resulting residual possess highest Gaussian property possible.

3.4.2 *Image noise comparison: Statistical Model selection*

The success of this Gaussian residual optimization depends on the matrix used to compare the residual noise from particle image and the ideal noise (Gaussian noise) statistical profile. The choice of choosing the most effective statistical model to evaluate the distribution profile of residual image is very important for the optimization process to be effective.

A crucial tool in evaluating residual image whether it conforms to Gaussian distribution or not is known as a goodness-of-fit test. A well performed goodness-of-fit test should be able to quantify the discrepancy between observations and the statistical model that it aims to quantify. Several goodness-of-fit are evaluate on noisy image, including Chi-Square test [39], Kolmogorov-Smirnov test [40], Anderson-Darling test [41], and Jarque-Bera test [42]. Each of the statistical model are tested by comparing the numerical result from different Region-Of-Interests (ROI) in the same noisy distribution. Ideally, regardless of the ROI in image, the results should be the same. The visual representation of 2 different ROI from same noisy image is illustrated in Figure 3.14.



Figure 3.14 The image of 5 percent Gaussian noise. The blue circle separates two ROI, the inside of circle and the outside of circle for comparison.

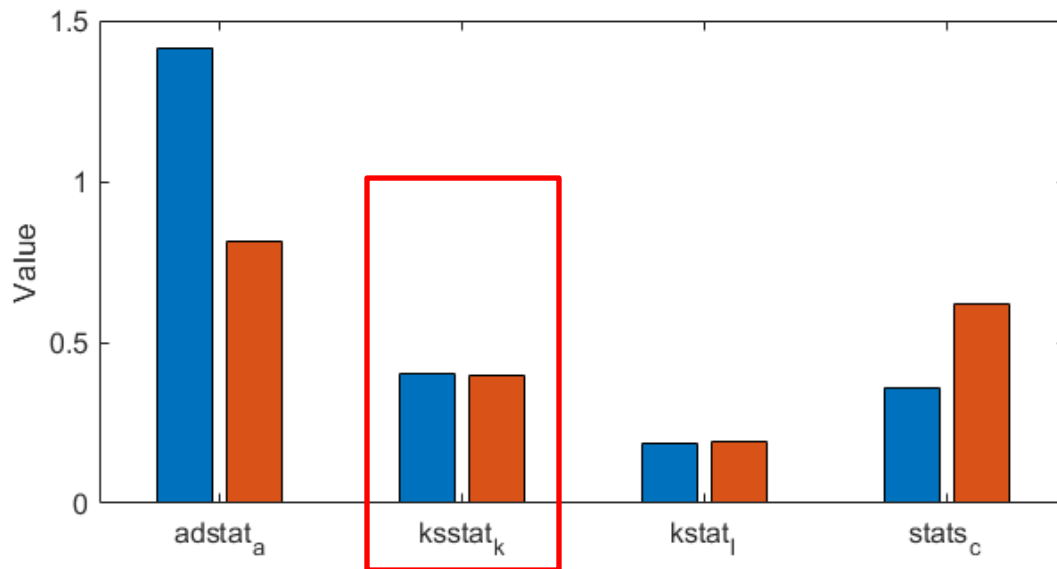


Figure 3.15 The value-pair of each statistical tests.

According to the evaluation matrix shown in Figure 3.15, Kolmogorov-Smirnov test ($ksstat$) yields consistent results in determining whether the residual image follows Gaussian distribution. Whereas the other statistical model gives inconsistent results from the same noisy residual image. Therefore, the other statistical model that give different results from the same noisy image cannot be used to compare the noise in the cost function.

3.4.3 *Cost Function*

The end goal of residual noise optimization is to iteratively generate synthetic particle image which most resemble the true image. The method to quantify residual image has already been described in previous section. The generated synthetic image is then subtracted from real image to get residual image. The residual image is then compared, using statistical model, with Gaussian noise model. During optimization process, the X, Y coordinates are iteratively updated to minimize the difference of Kolmogorov-Smirnov test between residual image and ideal Gaussian distribution. We used the particle-swarm global optimization method through MATLAB function *particleswarm* to adjust all particle locations simultaneously until the residual is minimum [43].

3.5 RANDOM PARTICLE RADIUS IDENTIFICATION TEST

The particle image identification through the point spread function as described in Chapter 2.3.2 requires that the image inversion relationship between the particle image and point light source has to be constructed using a single particle radius value. However, in experimental particle image, the radius of particle is not always consistent.

In order to assess the robustness of particle identification performance, several tests on variable particle radius are performed at varying particle density. At the first step, the particle image is separate into image blob of connected particles. Then the Modified Cascade Cross Correlation Method (MCCM) described in Chapter 2.3.1 is used to determine the approximate radius of particle in each blob. Since each particle in each blob does not have the same radius, the algorithm reject particle that falls in the first quartile and last quartile. The radius values that fall in to the second quartile and third quartile are averaged and use as σ for weight matrix construction. Then the weight matrix is constructed for PSF algorithm to calculate the location of each particle.

The experiments conducted studies on 2 separate range of particle radius, 0.6 to 1.4 pixels and 1 to 1.2 pixels. For each range of particle size, the particle density is varied from 0.01 to 0.10 particle per pixel. The effect of image noise is also tested, the Gaussian noise of level 5 percent is added to the particle image. Then the RDUNet denoising architecture described in Chapter 2.5.2 is used to denoise the corrupted image.

Chapter 4. RESULTS

In this chapter, the results from each of the method described in Chapter 3 are presented. The result obtained from each individual method are not quantitatively comparable in terms of performance metrices since they are suitable for different particle configurations.

4.1 CRITICAL PARAMETERS SELECTION

4.1.1 *Distance Transform rejection threshold*

The optimum value α_{optim} for minimum score associated to each detection is determined by calculating the yield and reliability performance matrices while varying the cutoff threshold. The procedure is performed by varying α from 0 to 1.2. The particle image at size 32 by 32 pixel with the value of N_{ppp} at 0.10 and 5 percent noise level are chosen for this experiment. The result of this experiment is shown in Figure 4.1.

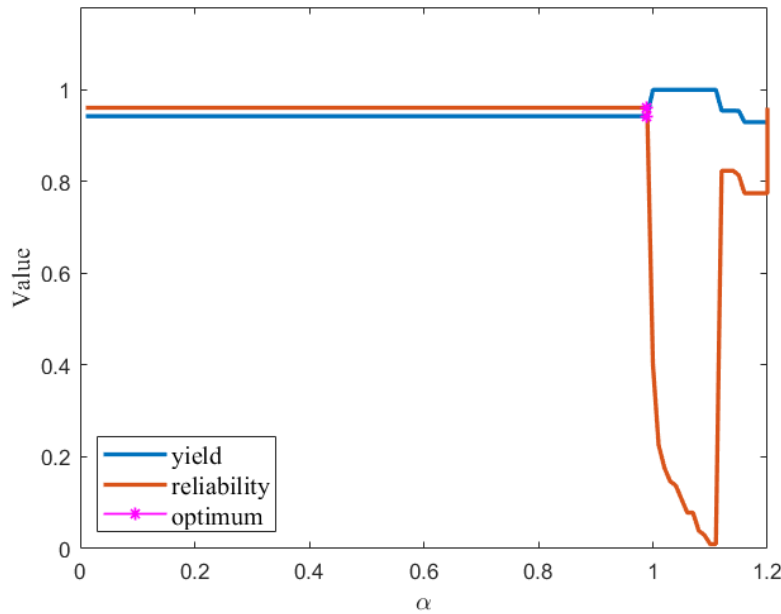


Figure 4.1 The plot of yield and reliability performance matrices while varying α parameters.

The vertical magenta line indicates the optimum value of α chosen for the workflow.

With a sharp decline of the yield value and sharp increase in reliability value at $\alpha = 0.99$, the value of α is selected to use throughout the workflow.

4.1.2 Density Based Spatial Clustering thresholds

The performance matrices yield, and reliability are calculated while the critical value in MATLAB *dbscan* function, is varied from 0 to 1.2 pixel. The particle image at size 32 by 32 pixel with the value of N_{ppp} at 0.10 and 5 percent noise level are chosen for this experiment. The result of this experiment is shown in Figure 4.2.

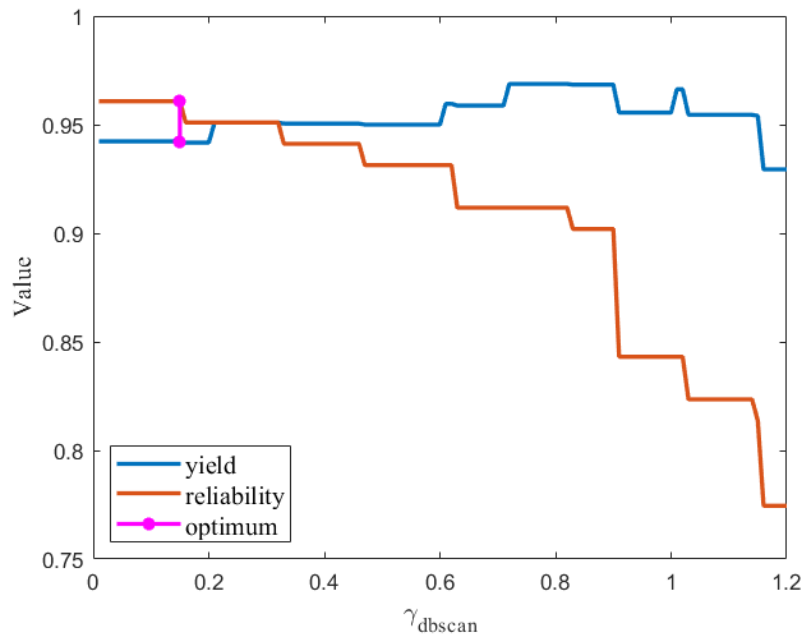


Figure 4.2 The plot of yield and reliability performance matrices while varying γ_{dbscan} parameters. The vertical magenta line indicates the optimum value of γ_{dbscan} chosen.

As expected, as the distance threshold γ_{dbscan} increases the yield value dropped because the algorithm is more inclined to reject even particle that located far apart. In contrast to the yield, the value of reliability increases at lower half of γ_{dbscan} range. This is because less yield value means

that with less detection, the likelihood of getting legitimate detection increases. A good compromise between the yield and reliability matrices is located at γ_{dbscan} equals to 0.15. This value is used throughout the experiment.

4.2 NUMERICAL RESULTS

4.2.1 Particle Identification Performance at 5 percent Gaussian noise

For the experimental procedure described in the Chapter 3, the mean value of the particle localization error (y-axis) with respect to ground truth position are plotted in Figure 4.3 against the varying values of N_{ppp} (x-axis). The performance of proposed particle identification workflow is evaluated at fixed value of 5 percent Gaussian noise.

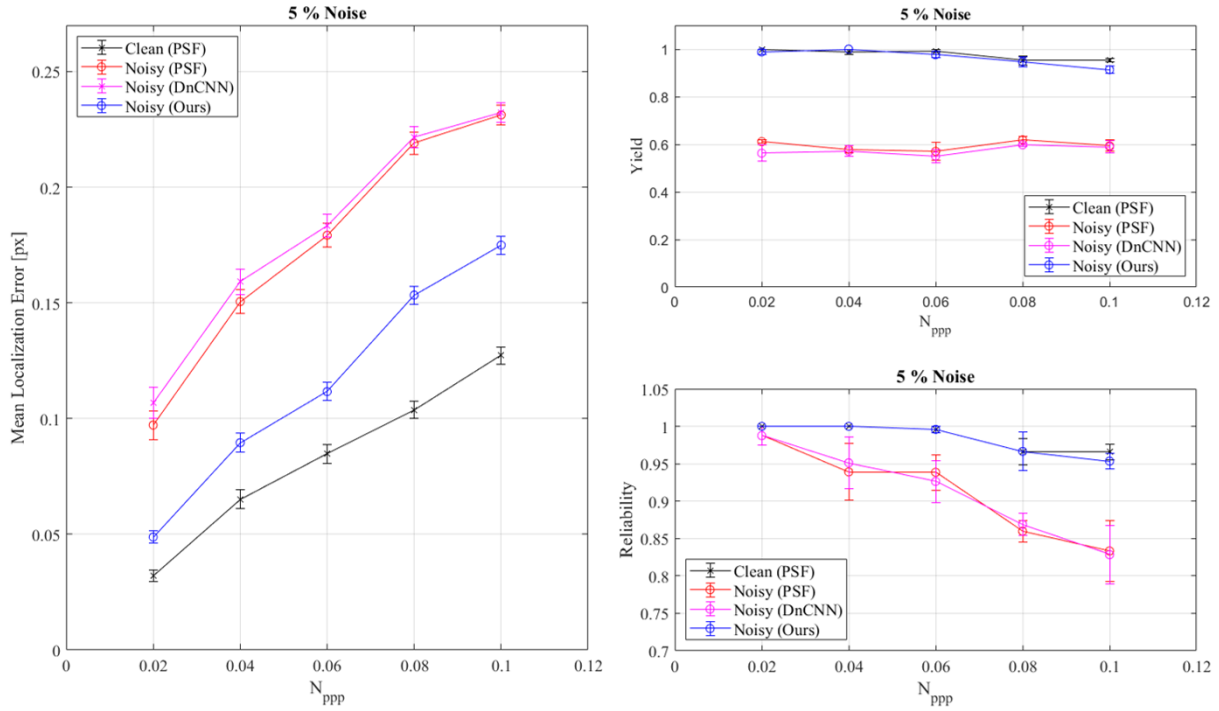


Figure 4.3 Comparison between Clean image using PSF (Black), Noisy image using PSF (Red), Noisy image denoised using DnCNN (Magenta), and Noisy image denoised with RDUNet (Blue) in terms of Mean localization error, yield, and reliability matrices. Regardless of denoising techniques, all methods subjected to our proposed False Positive (FP) rejection workflow.

With regards to the particle identification performance, the Mean Localization Error for noisy image and the de-noised image through DnCNN denoising algorithm, are almost the same. By keeping the clean image as a baseline for comparison, the experimental results shows that the de-noised image using RDUNet algorithm enables PSF particle image recovery to be the most accurate among the four. At the noise level of 5 percent, the Mean Localization Error value is decreased by 24.7 percent for the case of N_{ppp} at 0.10. Considering the baseline Mean Localization Error for clean image, the increase in Mean Localization Error value was cut by 54 percent.

As expected, the de-noised image through DnCNN denoising algorithm even demonstrated higher value in mean localization error due to distortion of the original content due to denoising algorithm. It can be said that the performance of denoising algorithm to use in particle tracking application is very crucial in achieving accurate particle location when the particle image is subjected to noise.

In terms of yield, the yield increased 57 percent for our proposed workflow in comparison to the PSF method without particle rejection algorithm. When comparing to clean image, the yield value only decreased by 4.2 percent. In case of the original PSF method, the yield value decreased by 39 percent.

In terms of reliability, the reliability increased 15 percent for our proposed workflow in comparison to the PSF method without particle rejection algorithm. When comparing to clean image, the reliability value only decreased by 1.2 percent. In case of the original PSF method, the reliability value decreased by 14 percent.

4.2.2 Robustness of Particle Identification Performance at different noise level

The robustness of particle identification algorithm against unknown noise level from 1 to 5 percent Gaussian noise are shown in the Figure 4.4, Figure 4.5, and Figure 4.6.

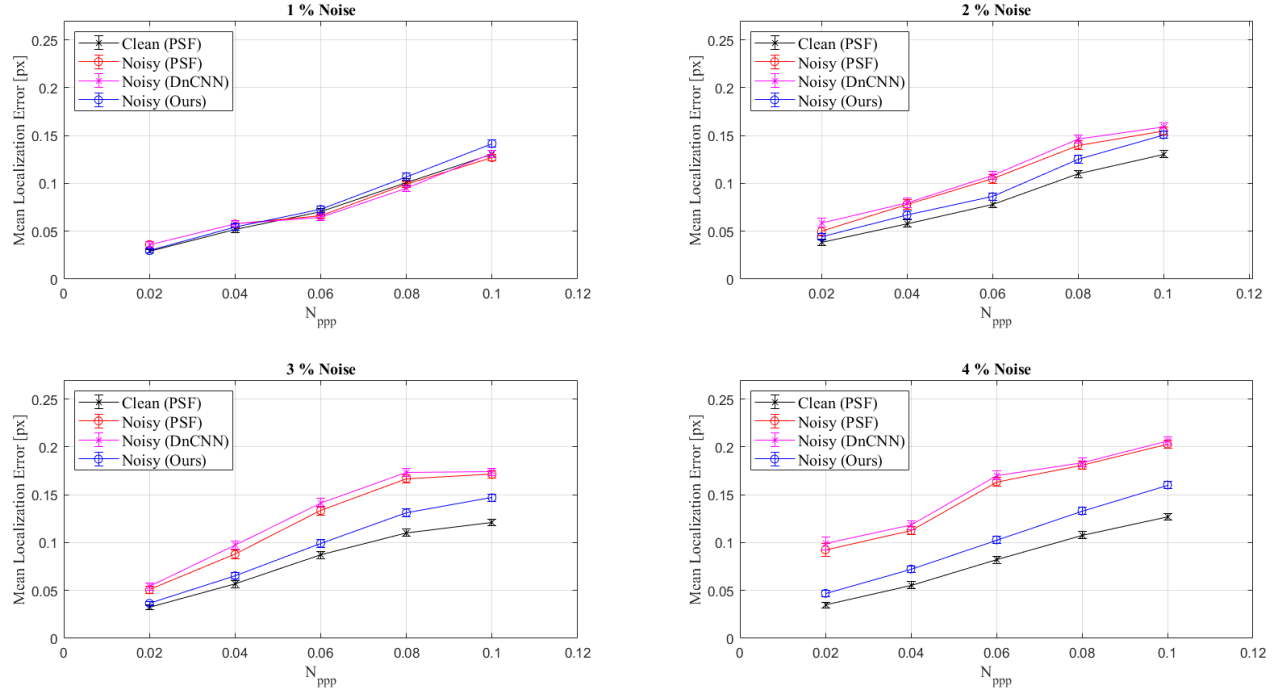


Figure 4.4 The Mean Localization Error at different noise level from 1 to 4 percent.

As anticipated, the increase in strength of noise from 1 to 4 percent leads to an increase in Mean Localization Error respectively. At the 1 percent noise level, which is the lowest noise level tested, the difference between the clean image and the noisy image in terms of mean localization error is relatively low compared to higher noise level. Therefore, as the noise level increases, the improvement on localization accuracy has greater effect. Without the information about the noise level need to be known beforehand, the proposed workflow always performs better than the results without it.

The detailed results for all configurations are described in Table 4.6, Table 4.7, and Table 4.8.

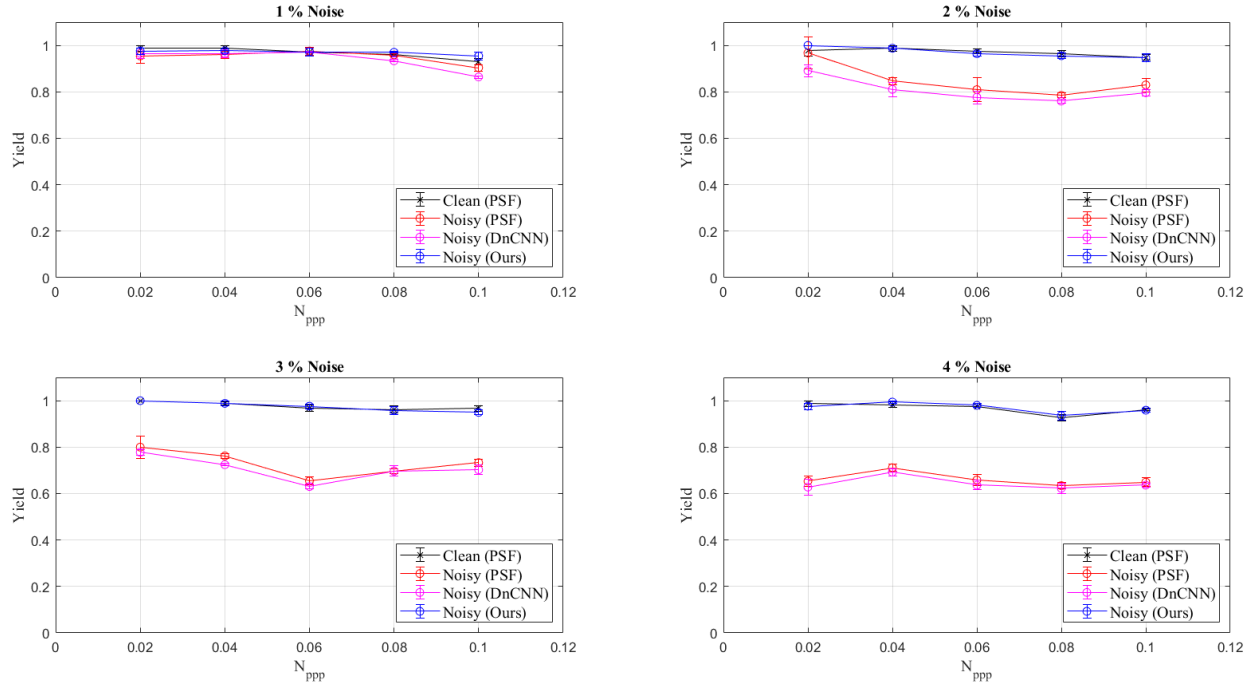


Figure 4.5 The yield at different noise level from 1 to 4 percent Gaussian noise.

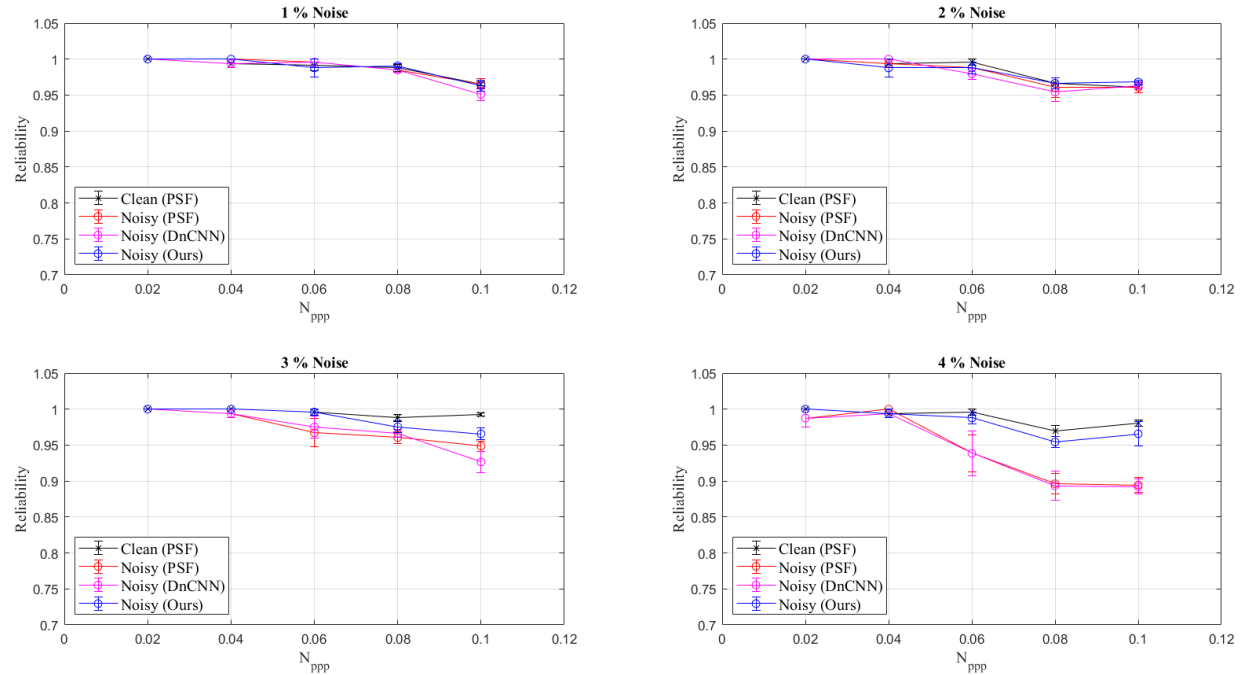


Figure 4.6 The reliability at different noise level from 1 to 4 percent Gaussian noise.

In terms of yield and reliability, the behaviors are similar to Mean Localization Error.

Table 4.6. The Mean Localization Error at different noise level for each technique.

Noise Level	N_{ppp}	Clean (PSF)	Clean (PSF) SE	Noisy (PSF)	Noisy (PSF) SE	Noisy (DnCN N)	Noisy (DnCN N) SE	Noisy (Ours)	Noisy (Ours) SE
1.0	0.020	0.029	0.002	0.036	0.003	0.036	0.003	0.030	0.002
1.0	0.040	0.052	0.003	0.058	0.004	0.058	0.004	0.055	0.003
1.0	0.060	0.070	0.003	0.066	0.003	0.065	0.003	0.073	0.003
1.0	0.080	0.101	0.004	0.099	0.004	0.095	0.003	0.107	0.004
1.0	0.100	0.131	0.004	0.127	0.004	0.131	0.004	0.142	0.004
2.0	0.020	0.038	0.003	0.050	0.004	0.059	0.005	0.044	0.004
2.0	0.040	0.058	0.004	0.078	0.005	0.080	0.005	0.067	0.004
2.0	0.060	0.078	0.004	0.105	0.004	0.108	0.004	0.086	0.004
2.0	0.080	0.110	0.004	0.140	0.004	0.147	0.004	0.126	0.004
2.0	0.100	0.131	0.004	0.155	0.004	0.159	0.004	0.151	0.004
3.0	0.020	0.033	0.002	0.051	0.004	0.054	0.004	0.037	0.002
3.0	0.040	0.057	0.004	0.088	0.004	0.098	0.004	0.065	0.004
3.0	0.060	0.087	0.004	0.133	0.005	0.142	0.005	0.099	0.004
3.0	0.080	0.110	0.004	0.167	0.004	0.173	0.004	0.131	0.004
3.0	0.100	0.121	0.003	0.171	0.004	0.174	0.004	0.147	0.004
4.0	0.020	0.035	0.003	0.092	0.007	0.099	0.007	0.047	0.003
4.0	0.040	0.056	0.004	0.113	0.005	0.118	0.005	0.072	0.003
4.0	0.060	0.082	0.004	0.163	0.005	0.170	0.005	0.103	0.004
4.0	0.080	0.108	0.004	0.181	0.004	0.184	0.004	0.133	0.004
4.0	0.100	0.127	0.004	0.203	0.004	0.206	0.004	0.160	0.004
5.0	0.020	0.032	0.003	0.097	0.006	0.107	0.007	0.049	0.003
5.0	0.040	0.065	0.004	0.151	0.005	0.159	0.005	0.090	0.004
5.0	0.060	0.085	0.004	0.179	0.005	0.183	0.005	0.112	0.004
5.0	0.080	0.104	0.004	0.219	0.005	0.222	0.005	0.153	0.004
5.0	0.100	0.127	0.004	0.231	0.004	0.232	0.004	0.175	0.004

*SE = Standard Error

Table 4.7. The Yield at different noise level for each technique.

Noise Level	N_{ppp}	Clean (PSF)	Clean (PSF) SE	Noisy (PSF)	Noisy (PSF) SE	Noisy (DnCN N)	Noisy (DnCN N) SE	Noisy (Ours)	Noisy (Ours) SE
1.0	0.020	0.988	0.012	0.955	0.026	0.965	0.022	0.976	0.014
1.0	0.040	0.988	0.012	0.960	0.016	0.966	0.019	0.976	0.009
1.0	0.060	0.984	0.011	0.976	0.004	0.972	0.004	0.972	0.018
1.0	0.080	0.970	0.006	0.959	0.011	0.934	0.005	0.970	0.003
1.0	0.100	0.941	0.014	0.904	0.004	0.864	0.005	0.955	0.018
2.0	0.020	0.988	0.012	0.967	0.033	0.891	0.026	1.000	0.000
2.0	0.040	0.988	0.012	0.846	0.016	0.811	0.026	0.988	0.007
2.0	0.060	0.976	0.010	0.810	0.039	0.777	0.028	0.964	0.012
2.0	0.080	0.967	0.012	0.785	0.014	0.760	0.010	0.955	0.013
2.0	0.100	0.952	0.011	0.830	0.017	0.796	0.012	0.946	0.017
3.0	0.020	1.000	0.000	0.800	0.000	0.777	0.014	1.000	0.000
3.0	0.040	0.994	0.006	0.762	0.009	0.725	0.003	0.988	0.012
3.0	0.060	0.968	0.014	0.654	0.018	0.630	0.010	0.976	0.010
3.0	0.080	0.967	0.010	0.696	0.013	0.697	0.020	0.959	0.017
3.0	0.100	0.974	0.010	0.733	0.022	0.702	0.019	0.949	0.010
4.0	0.020	0.988	0.012	0.654	0.053	0.628	0.036	0.976	0.014
4.0	0.040	0.988	0.007	0.711	0.017	0.692	0.018	0.994	0.006
4.0	0.060	0.976	0.005	0.658	0.015	0.636	0.016	0.980	0.008
4.0	0.080	0.936	0.011	0.634	0.018	0.623	0.026	0.935	0.019
4.0	0.100	0.964	0.002	0.649	0.016	0.636	0.012	0.959	0.009
5.0	0.020	1.000	0.000	0.611	0.028	0.563	0.036	0.988	0.012
5.0	0.040	0.988	0.012	0.579	0.022	0.571	0.020	1.000	0.000
5.0	0.060	0.992	0.005	0.571	0.024	0.551	0.026	0.976	0.013
5.0	0.080	0.961	0.011	0.619	0.011	0.598	0.009	0.947	0.020
5.0	0.100	0.959	0.007	0.594	0.028	0.590	0.024	0.913	0.015

*SE = Standard Error

Table 4.8. The Reliability at different noise level for each technique.

Noise Level	N_{ppp}	Clean (PSF)	Clean (PSF) SE	Noisy (PSF)	Noisy (PSF) SE	Noisy (DnCN N)	Noisy (DnCN N) SE	Noisy (Ours)	Noisy (Ours) SE
1.0	0.020	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
1.0	0.040	0.994	0.006	1.000	0.000	0.994	0.006	1.000	0.000
1.0	0.060	0.992	0.008	0.996	0.004	0.996	0.004	0.988	0.012
1.0	0.080	0.988	0.005	0.985	0.003	0.985	0.003	0.991	0.003
1.0	0.100	0.963	0.005	0.966	0.006	0.951	0.011	0.963	0.007
2.0	0.020	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
2.0	0.040	0.994	0.006	0.994	0.006	1.000	0.000	0.988	0.012
2.0	0.060	0.996	0.004	0.988	0.008	0.980	0.008	0.988	0.008
2.0	0.080	0.966	0.008	0.960	0.014	0.954	0.013	0.966	0.008
2.0	0.100	0.958	0.008	0.961	0.008	0.963	0.002	0.968	0.002
3.0	0.020	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
3.0	0.040	1.000	0.000	0.994	0.006	0.994	0.006	1.000	0.000
3.0	0.060	0.992	0.008	0.967	0.018	0.975	0.014	0.996	0.004
3.0	0.080	0.985	0.006	0.960	0.009	0.966	0.006	0.976	0.009
3.0	0.100	0.988	0.002	0.949	0.005	0.926	0.016	0.966	0.008
4.0	0.020	1.000	0.000	0.988	0.013	0.988	0.013	1.000	0.000
4.0	0.040	0.994	0.006	1.000	0.000	0.994	0.006	0.994	0.006
4.0	0.060	0.996	0.004	0.939	0.024	0.939	0.029	0.988	0.008
4.0	0.080	0.970	0.008	0.896	0.018	0.893	0.022	0.954	0.008
4.0	0.100	0.980	0.004	0.895	0.007	0.892	0.012	0.966	0.017
5.0	0.020	1.000	0.000	0.988	0.013	0.988	0.013	1.000	0.000
5.0	0.040	1.000	0.000	0.939	0.038	0.951	0.034	1.000	0.000
5.0	0.060	0.996	0.004	0.939	0.024	0.926	0.025	0.996	0.004
5.0	0.080	0.966	0.018	0.860	0.015	0.869	0.015	0.966	0.026
5.0	0.100	0.966	0.010	0.833	0.041	0.828	0.038	0.953	0.011

*SE = Standard Error

4.2.3 Robustness of Particle Identification workflow at large image size

In order to evaluate the performance of the proposed workflow for large particle image size, the synthetic particle image of size 160 by 160 pixels is tested (Figure 4.7). The N_{ppp} is 0.10 particle per pixel and the Gaussian noise level is 5 percent.

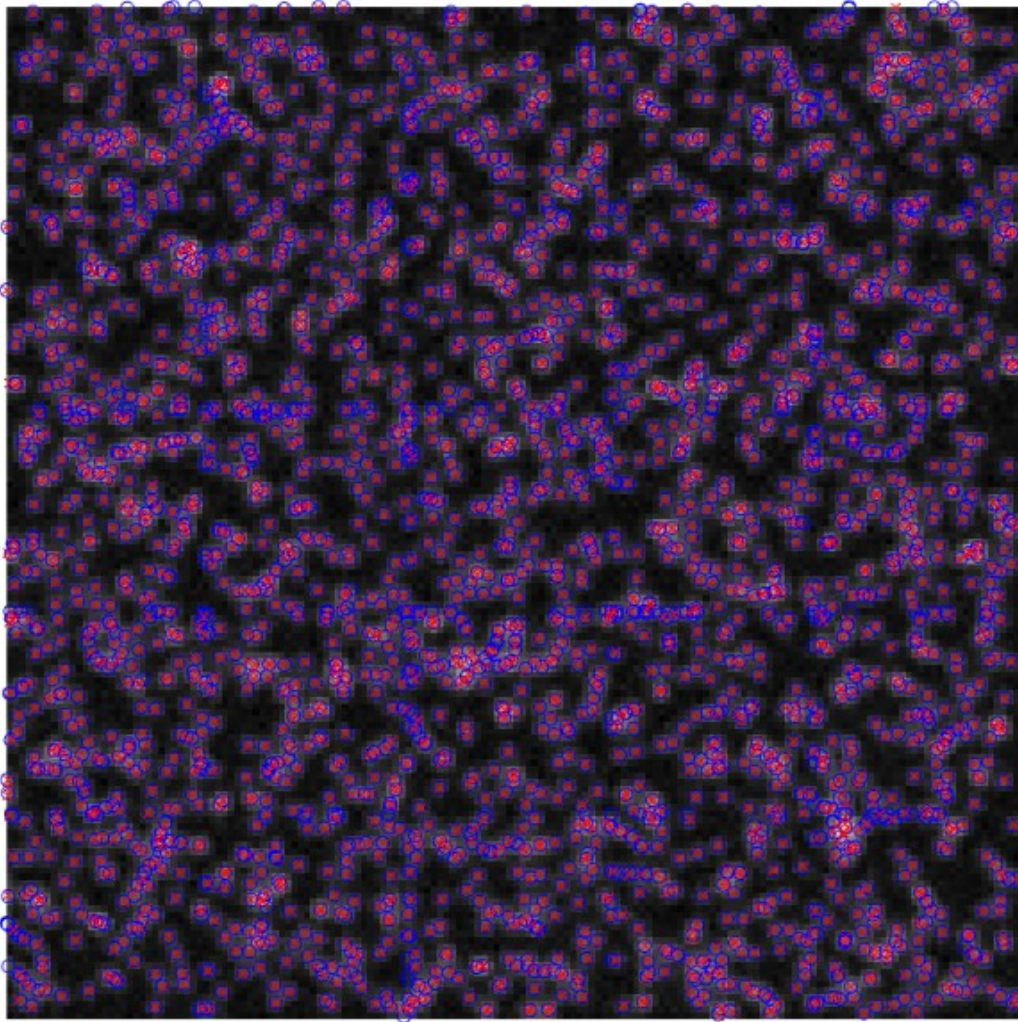


Figure 4.7 The detection results for the 160 by 160 pixels image at $N_{ppp} = 0.10$ and 5% Gaussian noise. The ground truth location (red cross) and detections (blue circle) are superimposed over the particle image.

Since the original particle image are separated into several image patch of size 32 by 32 pixels, the result of particle reconstruction through PSF is marginally different from processing smaller image with the size equals to the patch itself.

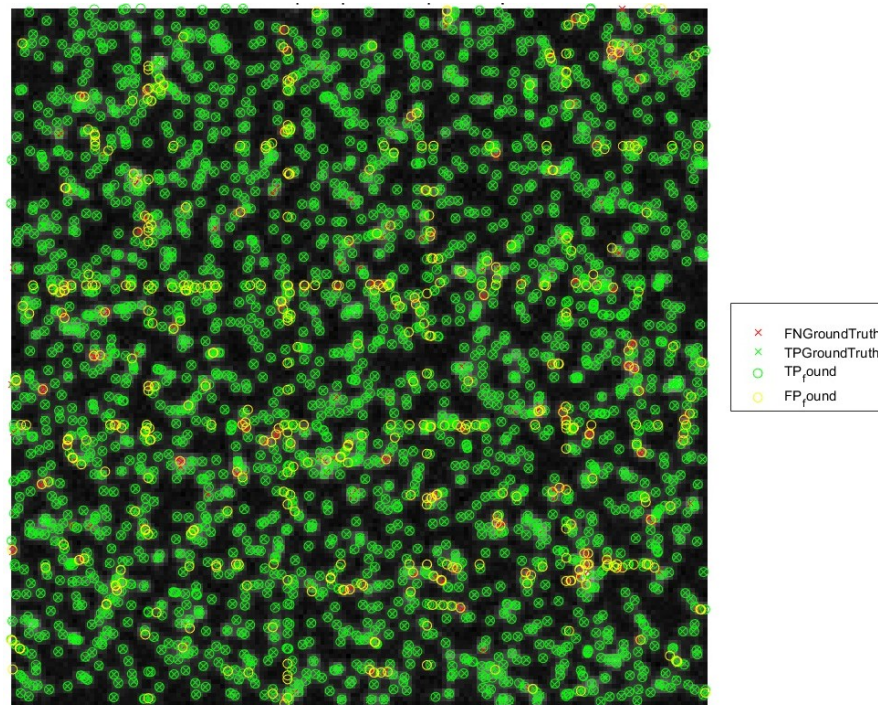


Figure 4.8 The detection of proposed workflow evaluated in terms of True Positive (TP), False Positive (FP) and False Negative (FN).

The obtained result, as illustrated in Figure 4.8, shows a significant increase in the False Positive (FP) indicated in the yellow circle in close proximity to the region where particle image is separated into image patches. This could be resulted from the duplicate detection of the same particle separated into different image patch. According to the Figure 4.8, the amount of False Negative is not affected by the separation of image as the locations of False Negative are evenly distributed throughout the image. This aspect of the problem still requires further investigation in

terms of reduction of False Positive (FP) near the edge of each image patch. The behavior for this is described in Figure 4.9.

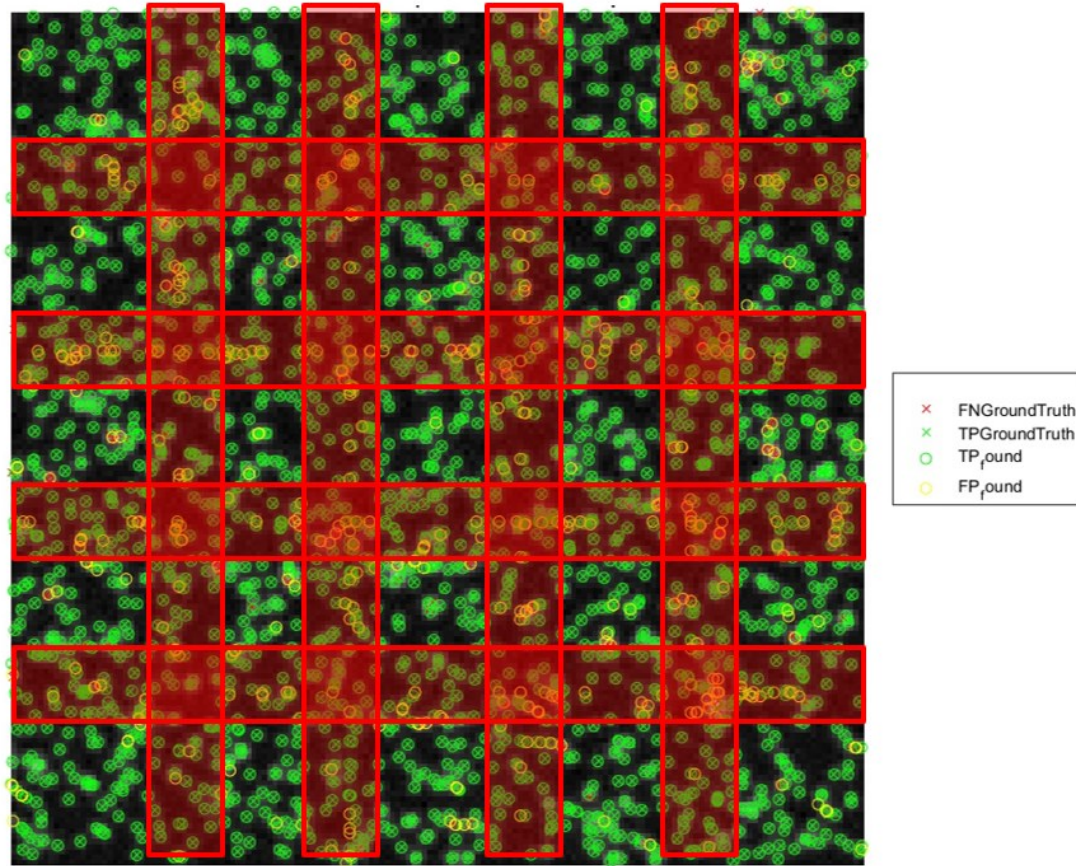


Figure 4.9 The regions where individual image patches are merged indicated in red rectangular box. The number of False Positive detections (FP, yellow) in the indicated regions clearly increased.

Table 4.9. Particle Identification and Localization Performance for large-scale image.

Parameter	Value	Standard Deviation
Mean Localization Error	0.1909	0.1227
Yield	0.8205	-
Reliability	0.9109	-

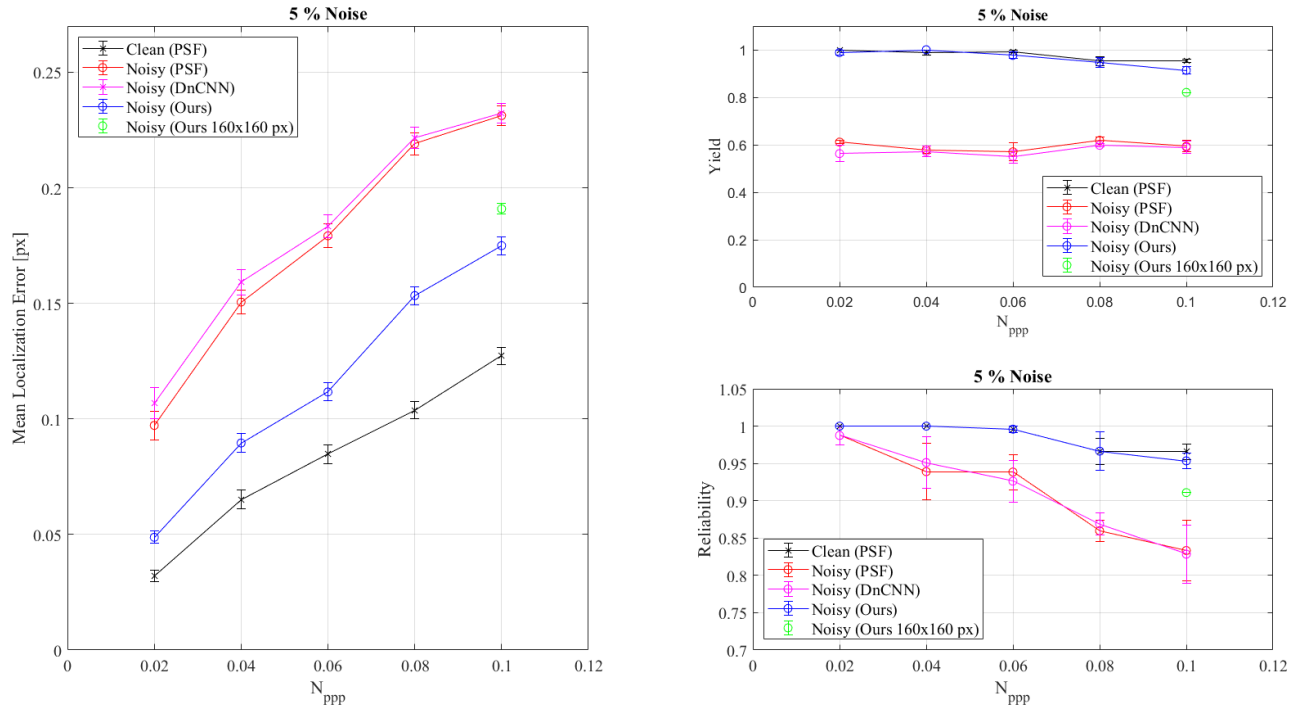


Figure 4.10 The 160 by 160 pixel particle image performance (Green) in comparison to small image patch 32 by 32 pixel.

The comparison between 32 by 32 image patch and 160 by 160 particle image, as presented in Figure 4.10., there is noticeable upsurge of Mean Localization Error, drop in Yield and Reliability. However, the performance is still incomparably better than PSF method. The author believes that the reason for this drop in performance is due to the inconsistency of particle detection near the edge of each image patch. The approach that could be implement in order to solve this problem is described in the future work section.

4.3 COMPUTATIONAL TIME

The operating system of the workstation used was Windows 10 64-bit with AMD Ryzen 9 3950X Processor, Nvidia RTX 3060 GPU, and 32 GB of DDR4 system memory. The particle identification workflow proposed in Chapter 3.1 time consumption increases as the size of the input image increases. Also, with the increase in N_{ppp} , the computation time, as illustrated in Figure 4.11, also increases. According to the results, it can be presumed that due to the higher number of overlapping particles, the complexity of point spread inversion calculation, the time efficiency of each image patch degraded proportionally.

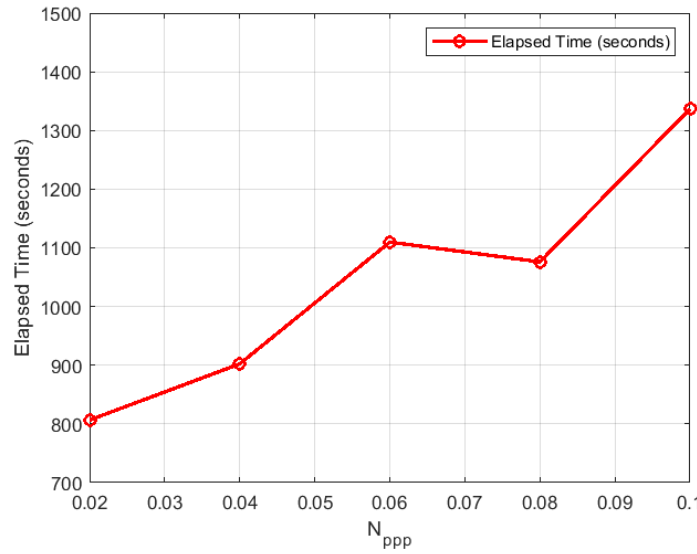


Figure 4.11 The plot of elapsed time for each image patch (32 by 32 pixels) against the particle density N_{ppp} .

In case of larger image than 32 by 32 pixel. For instance, 160 by 160 pixel, the computational time of the whole image would increase 25 times for the same N_{ppp} . Because of the nature of patch based image processing, the workflow can be done using distributed parallel computing to run the processing simultaneously, speed up the time required to process the particle image.

4.4 PARTICLE RECONSTRUCTION RESULTS

Since the limitation of the method described in the Section 3.4 is that the particle cannot be overlapped, the result of this algorithm is separated from the main proposed workflow.

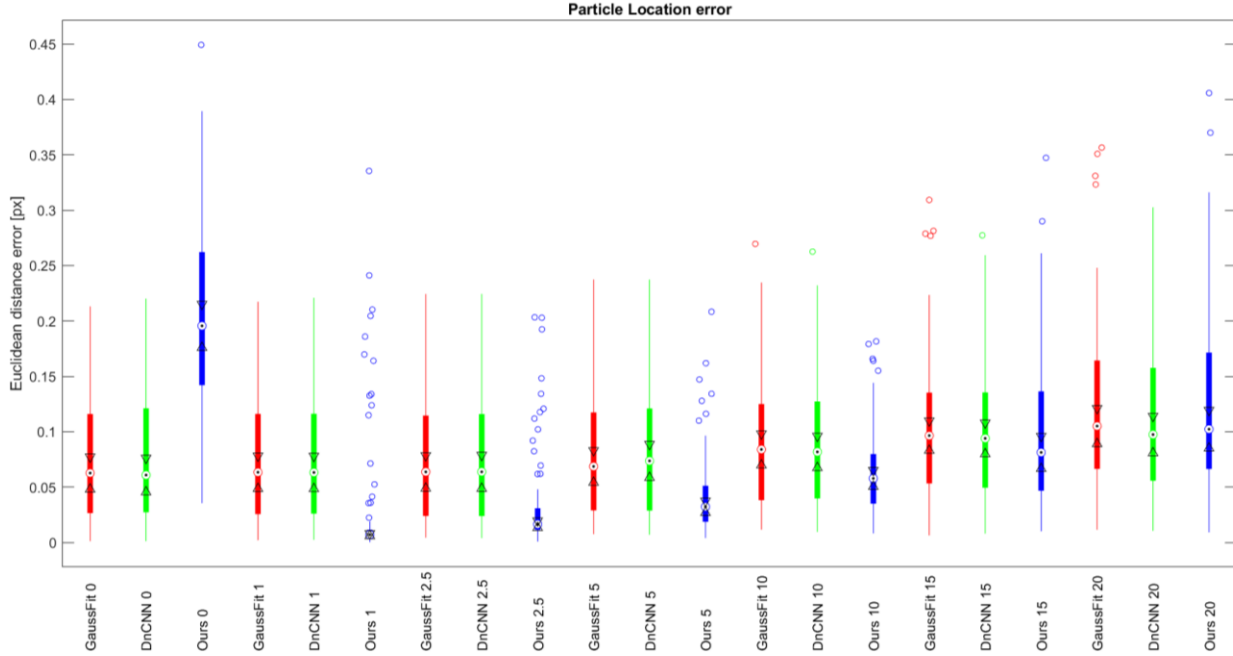


Figure 4.12 The boxplot representation of particle location error of using only 2D Gaussian function fit (Red), denoise image using DnCNN then 2D Gaussian fit (Green), our proposed iterative particle reconstruction and residual noise comparison (Blue). The number associated with each data refers to the Gaussian noise level in percent.

According to the results shown in Figure 4.12, this method is improving the particle location accuracy at the Gaussian noise level lower than 15 percent. At the higher Gaussian noise level, the particle location accuracy is not better than using only 2D Gaussian fit.

Moreover, this method cannot process overlapping particle image due to the calculation of initial particle location that calculated from the centroid of each connected region generated from U-Net.

4.5 RANDOM PARTICLE RADIUS RESULTS

4.5.1 Random size from 1 to 1.2 pixel

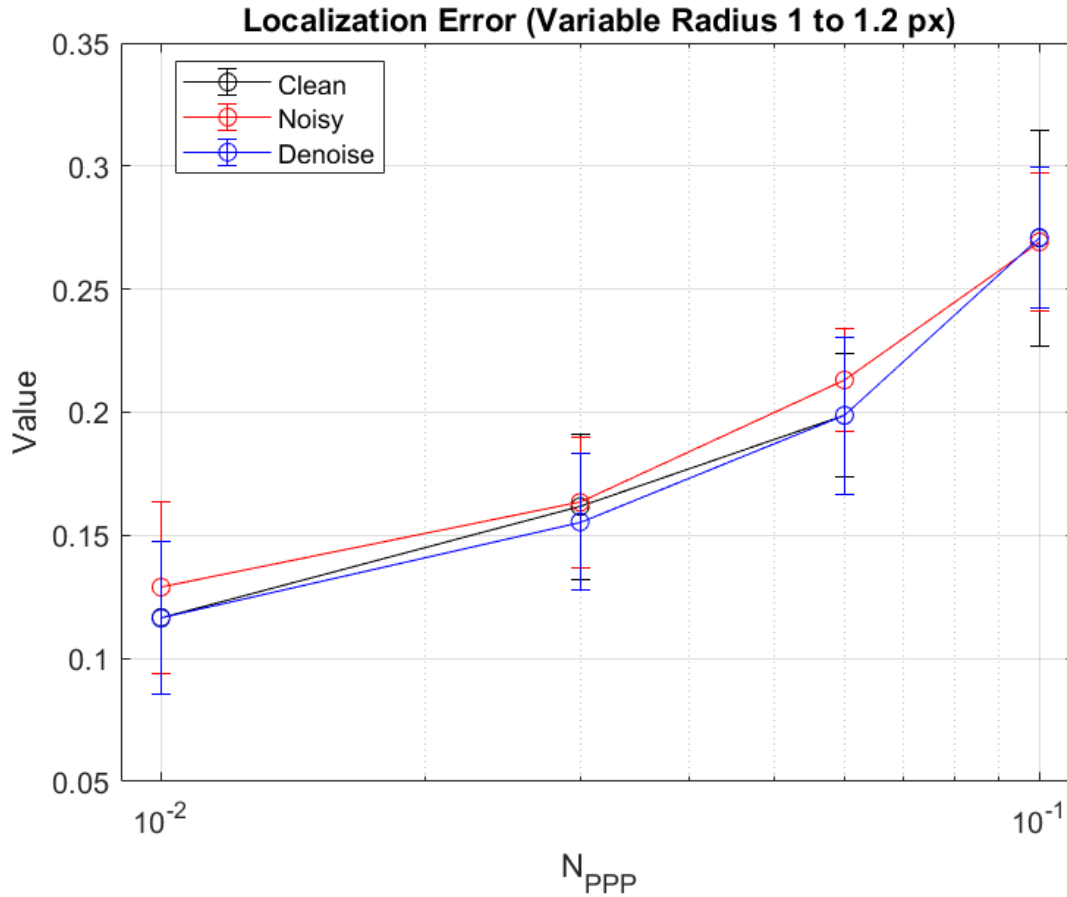


Figure 4.13 The localization performance for varying radius particle (1 to 1.2 pixel).

In terms of localization accuracy of varying radius from 1 to 1.2 pixel particle, the results regardless of clean, noisy, or using denoising filters give no significant difference between them. As the particle density increases, the error value increases as indicated in Figure 4.13.

According to the results shown in Figure 4.13, it could be concluded that the localization performances for varying particle radius are not due to the effect of added noise in image. In

comparison to the constant radius particle size, the construction of the Weight matrix for PSF particle localization using the approximated particle radius through MCCM method yield much greater positional error.

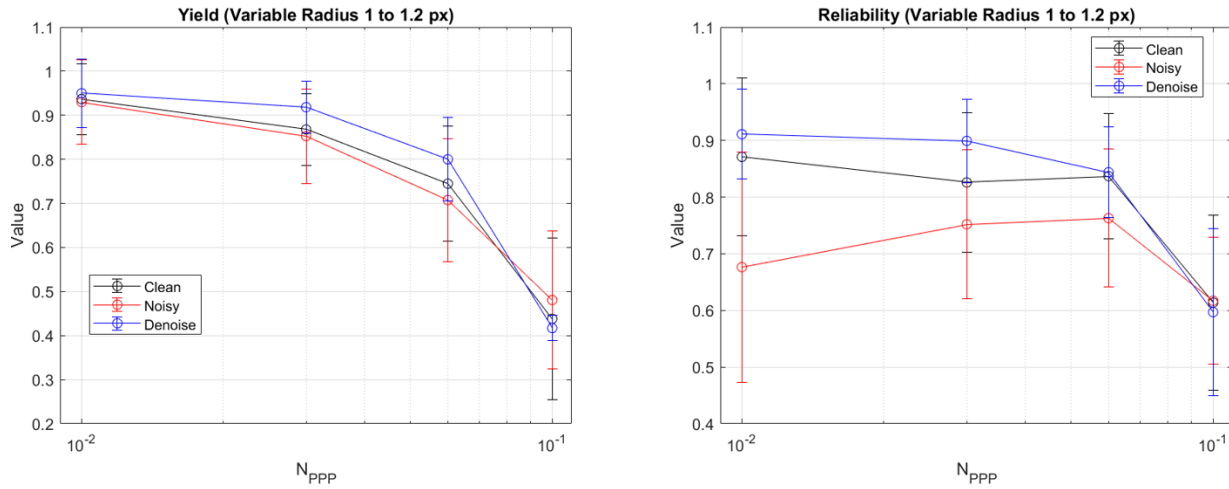


Figure 4.14 The Yield and Reliability performance of varying particle radius (1 to 1.2 pixel).

In terms of the Yield and the Reliability performance metrics (Figure 4.14), the effect of added image noise significantly lowered the Reliability across all particle density. Interestingly, the use of RDUNet denoising algorithms to remove noise from the particle image can lead to an improvement in the Reliability in lower half of tested particle density range compared to clean image. In terms of the Yield, the double in particle density resulted in lowering the Yield value by half. The 50 percent reduction in Yield cancel out the benefits of increasing particle density.

The experiment on varying particle radius also performed on the range of particle radius from 0.6 to 1.4 pixel. The performance metrics presented in the Figure 4.15 follow the same behavior as the smaller particle range.

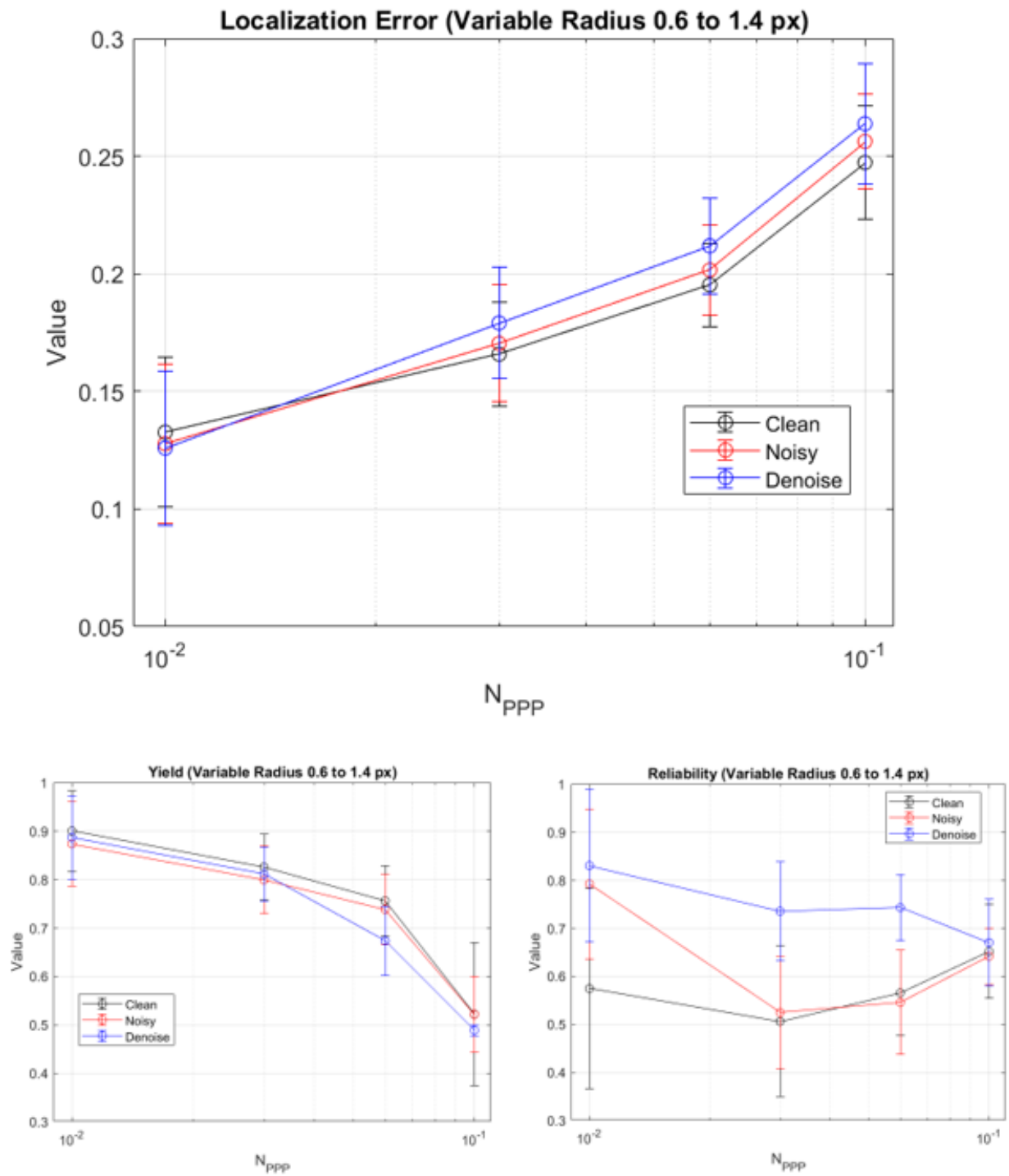


Figure 4.15 The Yield and Reliability performance of varying particle radius (0.6 to 1.4 pixel).

Chapter 5. FUTURE WORK

5.1 PERFORMANCE DROP MITIGATION ON LARGE IMAGE

For the reduction in False Positive detections where image patches are merged, the large particle image can be separated into 32 by 32 pixels image patches with some overlapping regions containing both patches. By considering detections from each image patch, the algorithm can be developed such that if each of the detection of the same particle from both patches are not further than a given distance, they can be merged into a single detection. The author strongly believes that by implementing this, the FP will be drastically reduced. Additionally, the author also believes that this implementation of overlapping image patches would exponentially increase the computational cost associated with a given size of particle image, because the same particle in both patches need to be calculate twice.

Chapter 6. CONCLUSION

This thesis examines various approach to develop a robust workflow to easily identify and localize particle image in presence of additive white Gaussian noise with greater particle localization accuracy. The aggregated mix of deep neural network image segmentation, image denoising, distance transform, and cutting-edge clustering technique empower the ability to identify and localize particles in the presence of noise better than ever. Also, the process is highly scalable in parallel computing due to the nature of individual patch-based image processing. By utilizing high performance parallel computing technology, the long processing time can be mitigated. Considering the particle localization through residual noise hypothesis model, the improvement in localization accuracy is significant at low noise. However, the method lacks the competence in overlapping particle situation. In terms of our proposed iterative particle reconstruction algorithm, the identification accuracies are better at lower noise level. For the variable radius particle size, the sharp reduction in Yield and reliability at higher particle density cancel out the benefits of increasing particle density. Future work can also investigate more on different shape of particle, and also using more complex noise model while training the deep neural network to match real experimental data.

BIBLIOGRAPHY

- [1] A. J. Smiths and T. . Lim, *Flow Visualization - Techniques and Examples*. 2010.
- [2] A. Singh, *Optic flow computation: a unified perspective*, vol. 3. IEEE computer society press Los Alamitos, 1991.
- [3] C. J. D. Pickering and N. A. Halliwell, “Laser speckle photography and particle image velocimetry: photographic film noise,” *Appl. Opt.*, vol. 23, no. 17, p. 2961, 1984, doi: 10.1364/ao.23.002961.
- [4] D. Dabiri and C. Pecora, *Particle tracking velocimetry*. IOP Publishing Bristol, 2020.
- [5] C. Pecora, “Particle Tracking Velocimetry: A Review,” University of Washington, 2018.
- [6] K. Ohmi and H. Y. Li, “Particle-tracking velocimetry with new algorithms,” *Meas. Sci. Technol.*, vol. 11, no. 6, pp. 603–616, 2000, doi: 10.1088/0957-0233/11/6/303.
- [7] J. M. Newby, A. M. Schaefer, P. T. Lee, M. G. Forest, and S. K. Lai, “Convolutional neural networks automate detection for tracking of submicron-scale particles in 2D and 3D,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 115, no. 36, pp. 9026–9031, 2018, doi: 10.1073/pnas.1804420115.
- [8] Y. Feng, J. Goree, and B. Liu, “Errors in particle tracking velocimetry with high-speed cameras,” *Rev. Sci. Instrum.*, vol. 82, no. 5, 2011, doi: 10.1063/1.3589267.
- [9] Y. C. Lei *et al.*, “A vision-based hybrid particle tracking velocimetry (PTV) technique using a modified cascade correlation peak-finding method,” *Exp. Fluids*, vol. 53, no. 5, pp. 1251–1268, 2012, doi: 10.1007/s00348-012-1357-6.
- [10] M. Marxen, P. E. Sullivan, M. R. Loewen, and B. Jähne, “Comparison of Gaussian particle center estimators and the achievable measurement density for particle tracking velocimetry,” *Exp. Fluids*, vol. 29, no. 2, pp. 145–153, 2000, doi: 10.1007/s003489900085.
- [11] G. E. Elsinga, F. Scarano, B. Wieneke, and B. W. Van Oudheusden, “Tomographic particle image velocimetry,” *Exp. Fluids*, vol. 41, no. 6, pp. 933–947, 2006, doi: 10.1007/s00348-006-0212-z.
- [12] A. Cheminet, J. Krawczynski, P. Druault, and P. Image, “Particle Image Reconstruction for particle detection in Adam Cheminet , Jean-François Krawczynski , Philippe Druault
To cite this version : HAL Id : hal-01927708 Particle Image Reconstruction for particle

- detection in PTV Manuscript version : Accepted Man,” 2019.
- [13] N. Otsu, “THRESHOLD SELECTION METHOD FROM GRAY-LEVEL HISTOGRAMS,” *IEEE Trans Syst Man Cybern*, vol. SMC-9, no. 1, pp. 62–66, 1979, doi: 10.1109/TSMC.1979.4310076.
 - [14] W. Weng and X. Zhu, “INet: Convolutional Networks for Biomedical Image Segmentation,” *IEEE Access*, vol. 9, pp. 16591–16603, 2021, doi: 10.1109/ACCESS.2021.3053408.
 - [15] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, “U-net and its variants for medical image segmentation: A review of theory and applications,” *IEEE Access*, 2021, doi: 10.1109/ACCESS.2021.3086020.
 - [16] N. Siddique, P. Sidike, C. Elkin, and V. Devabhaktuni, “U-Net and its variants for medical image segmentation: theory and applications.”
 - [17] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” Accessed: Jul. 17, 2021. [Online]. Available: <http://lmb.informatik.uni-freiburg.de/>.
 - [18] “Create U-Net layers for semantic segmentation - MATLAB unetLayers.” <https://www.mathworks.com/help/vision/ref/unetlayers.html> (accessed Jul. 29, 2021).
 - [19] “Semantic Segmentation of Multispectral Images Using Deep Learning - MATLAB & Simulink Example.” <https://www.mathworks.com/help/vision/ug/multispectral-semantic-segmentation-using-deep-learning.html> (accessed Aug. 08, 2021).
 - [20] M. Mafi, H. Martin, M. Cabrerizo, J. Andrian, A. Barreto, and M. Adjouadi, “A comprehensive survey on impulse and Gaussian denoising filters for digital images,” *Signal Processing*, vol. 157, pp. 236–260, Apr. 2019, doi: 10.1016/J.SIGPRO.2018.12.006.
 - [21] P. B. Alisha and G. S. K, “Image Denoising Techniques-An Overview,” vol. 11, no. 1, pp. 78–84, 2016, doi: 10.9790/2834-11117884.
 - [22] C. W. J. Beenakker and M. Büttiker, “Suppression of shot noise in metallic diffusive conductors,” *RAPID Commun. Phys. Rev. B*, vol. 46, pp. 15–1992.
 - [23] P. Kefer *et al.*, “Performance of deep learning restoration methods for the extraction of particle dynamics in noisy microscopy image sequences,” doi: 10.1101/2020.11.04.368928.

- [24] G. R. Easley and D. Labate, “Image Processing using Shearlets.”
- [25] C. Tian, L. Fei, W. Zheng, Y. Xu, W. Zuo, and C.-W. Lin, “Deep Learning on Image Denoising: An Overview.”
- [26] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, 2017, doi: 10.1109/TIP.2017.2662206.
- [27] V. Jain and H. S. Seung, “Natural Image Denoising with Convolutional Networks.”
- [28] J. Gurrola-Ramos, O. Dalmau, and T. E. Alarcon, “A Residual Dense U-net Neural Network for Image Denoising,” *IEEE Access*, pp. 1–14, 2021, doi: 10.1109/ACCESS.2021.3061062.
- [29] “Semantic Segmentation Using Dilated Convolutions - MATLAB & Simulink.” <https://www.mathworks.com/help/vision/ug/semantic-segmentation-using-dilated-convolutions.html> (accessed Jul. 31, 2021).
- [30] “GitHub - JavierGurrola/RDUNet: A Residual Dense U-net Neural Network for Image Denoising.” <https://github.com/JavierGurrola/RDUNet> (accessed Jul. 31, 2021).
- [31] “ONNX | Home.” <https://onnx.ai/> (accessed Jul. 31, 2021).
- [32] “Import pretrained ONNX network - MATLAB importONNXNetwork.” <https://www.mathworks.com/help/deeplearning/ref/importonnxnetwork.html> (accessed Jul. 31, 2021).
- [33] C. R. Maurer, R. Qi, and V. Raghavan, “A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 2, pp. 265–270, Feb. 2003, doi: 10.1109/TPAMI.2003.1177156.
- [34] “Distance transform of binary image - MATLAB bwdist.” <https://www.mathworks.com/help/images/ref/bwdist.html> (accessed Jul. 16, 2021).
- [35] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” 1996, Accessed: Jul. 27, 2021. [Online]. Available: www.aaai.org.
- [36] “Density-based spatial clustering of applications with noise (DBSCAN) - MATLAB dbscan.” <https://www.mathworks.com/help/stats/dbscan.html> (accessed Aug. 05, 2021).
- [37] P. Ruhnau, C. Guetter, T. Putze, and C. Schnörr, “A variational approach for particle

- tracking velocimetry,” *Meas. Sci. Technol.*, vol. 16, no. 7, pp. 1449–1458, 2005, doi: 10.1088/0957-0233/16/7/007.
- [38] A. Cheminet, J. F. Krawczynski, and P. Druault, “Particle image reconstruction for particle detection in particle tracking velocimetry,” *Meas. Sci. Technol.*, vol. 29, no. 12, 2018, doi: 10.1088/1361-6501/aae9c2.
- [39] “Chi-square goodness-of-fit test - MATLAB chi2gof.”
<https://www.mathworks.com/help/stats/chi2gof.html> (accessed Aug. 07, 2021).
- [40] “One-sample Kolmogorov-Smirnov test - MATLAB kstest.”
<https://www.mathworks.com/help/stats/kstest.html> (accessed Aug. 07, 2021).
- [41] “Anderson-Darling test - MATLAB adtest.”
<https://www.mathworks.com/help/stats/adtest.html> (accessed Aug. 07, 2021).
- [42] “Jarque-Bera test - MATLAB jbtest.” <https://www.mathworks.com/help/stats/jbtest.html> (accessed Aug. 07, 2021).
- [43] “Particle swarm optimization - MATLAB particleswarm.”
<https://www.mathworks.com/help/gads/particleswarm.html> (accessed Aug. 06, 2021).

