Clustering of Individual Entities Based on Common Features

by

Maryam Bagheri

A dissertation submitted to the Department of Mechanical Engineering

Cullen College of Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in Mechanical Engineering

Chair of Committee: Andrea Prosperetti

Co-Chair of Committee: Rodolfo Ostilla Monico

Committee Member: Peggy Lindner

Committee Member: Ralph Metcalfe

Committee Member: Di Yang

University of Houston

December 2021

Acknowledgements

Pursuing a Ph.D. is a difficult path. In pursuing this path, first and foremost, I would like to express my greatest gratitude to my advisor, Prof.Andrea Prosperetti, for his continuous support, guidance, ideas, and patience. I will always be grateful that he opened the way for me to work on this topic. He not only deepened my Fluid Mechanics knowledge, but also refined my research skills. It will forever be a great honor for me to have worked under the supervision of such a passionate, tireless, and thorough scientist.

I want to thank the rest of my dissertation committee, Prof. Ralph Metcalfe, Dr. Peggy Lindner, Dr. Di Yang and Dr. Rodolfo Ostilla Monico for their insightful comments and questions, which motivated me to widen my research from various perspectives. I especially would like to thank Dr. Peggy Lindner, who provided extensive guidance and insight in areas of this work where my knowledge was lacking.

I also would like to thank previous members of Prosperetti research group, especially Dr. Gedi Zhou and Dr. Yuhang Zhang, who provided me with useful insights of the data used in this work. I also wish to thank my friends Zeinab Zamani and Dr. Reza Mousavi for accompanying me on this special journey and forming invaluable memories along the way. Last but not the least, I would like to thank my parents and my sisters for supporting me spiritually throughout writing this thesis.

Abstract

Identification of clusters in spatial or other datasets is of interest in many applications including epidemiology, medical image processing, landscape ecology, criminology, archeology, astronomy and many other fields.

In the current work, we propose a general method for clustering individual entities on the basis of a common feature for both a two- and three-dimensional spatial region. Specifically, the method is demonstrated on a dataset obtained from the resolved simulation of falling particles in upward directed fluid flow. These simulations were conducted in a computational domain in the form of a parallelepiped with a square cross section and aspect ratio of 3. The boundary conditions on all six boundaries enforced periodicity. The particle feature on which clustering is based is the vertical velocity. The clusters identified group particles which have a velocity larger (in modulus) than a specified multiple of the standard deviation of the vertical velocity of all the particles in the domain.

The method starts by dividing the region of interest into cells. To capture clusters that extend over several cells use is made of "masks" including many cells. The location and size of the masks are randomly generated and their number is such that each cell of the domain has an approximately equal probability to be covered by a mask. Masks are labeled as interesting if they contain a sufficient number of particles with large velocities. Counting the number of times that each cell has been covered by an interesting mask, each cell is assigned a value which is analogous to the intensity value of an image pixel. By using a global threshold, the region is binarized into high intensity and low intensity cells. The high intensity cells are grouped into clusters by a method which integrates the region growing and region merging methods of digital image processing. The method is shown to work well properly accounting for the spatial periodicity of the data and to be able to track the clusters in time.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation addresses a particular instance of a general problem that is encountered in many contexts, namely the identification of clusters of different entities in a suitable space. The clustering is to be based on one (or more) specific attribute(s) of the individual entities and the space can be physical (e.g., a geographical space) or a general metric space, the notion of metric being necessary to appropriately define clusters. This type of study, which is called spatial clustering in the literature, may be undertaken for a wide variety of reasons. For example, in criminology, the aim is to detect and investigate crime 'hot spots' and other forms of localized offense patterns, victims, and offenders. Street robberies may be observed to cluster around bars or vandalism hot spots are often found at or near public transit stops. Archaeological artifacts may be found in specific locations, such as close to and with sight of coastal areas, which implies that the sea may have some symbolic or practical significance. In astronomy, the placement of galaxies in the cosmos by detecting star clusters is of interest. Recent studies in economics have drawn attention to the benefits of spatial clustering. For example, house prices decreasing or increasing in a specific location will help investigate the growth of a city. In the study of disease surveillance, certain health indicators for a group of settlements might correlate with their distance and

direction from possible sources of environmental risk. Examples are the effect of exposure to atmospheric nitrogen oxides on stroke deaths, the unusual spread of a disease over a short period of time in a particular geographical area and others. In ecological processes, the pattern of occurrence of species may be explained by environmental/habitat features, such as the pattern of trees in a forest. Finding areas that have similar land-cover types using satellite imagery is a related application of spatial clustering.

In our study, we propose a method to group the individual events or entities that happen to be spatially close to each other. While we have developed the method for a specific purpose, briefly described in section 4.7, its applicability is more general and can easily be extended to other situations. To introduce the topic, in this chapter we focus on digital image segmentation as an application of spatial clustering mainly because our methodology used to identify clusters is similar to the methods used in digital image segmentation. We then discuss how our method is compared with those of image segmentation. At the end of the chapter, we will explain the motivation of our study.

1.1 Digital Image Segmentation

An image may be described as a two-dimensional function, $f(x, y)$, where x and y are spatial (plane) coordinates. The amplitude of f at any pair of coordinates (x, y) is called the intensity or graylevel of the image at that point. The picture is called a digital image when x, y, and the intensity values of f are all finite, discrete numbers. The field of digital image processing refers to the use of a digital computer to process digital pictures through algorithms. Pixel is the most often used word to describe the components of a digital picture. Digital image processing methods started to be employed in medical imaging, remote Earth resource studies,

and astronomy in the late 1960s and early 1970s.

One of the most significant milestones in the use of image processing in medical diagnostics occurred in the early 1970s with the introduction of computerized axial tomography (CAT), often known as computerized tomography (CT) for short. X-rays are one of the earliest types of electromagnetic radiation employed in medical imaging. Some previous works in the field of medical imaging include: comparing and analyzing different segmentation techniques to isolate a brain tumor from the other regions of the brain using a set of 2D Magnetic Resonance (MR) images(Gajanayake *et al.*, 2009), proposing different segmentation techniques for detecting of white matter in the brain from functional Magnetic Resonance Imaging (fMRI) (Srinivasa *et al.*, 2008), presenting a combination of different segmentation methods to detect 3D brain tumors(Narendran *et al.*, 2012), illustrating an image segmentation method to detect brain tumor which extracts the exact position and shape, specially when the tumor morphological changes remain subtle, irregular and thus it is difficult to assess by clinical examination (Dhage *et al.*, 2015), and proposing a method for contactless palmprint recognition (Suzuki *et al.*, 2020).

Binarization and image segmentation are important steps in image analysis and processing. As an example, in medical imaging, images need to be divided into sets of pixels or image objects, also known as segments, for computer-aided diagnosis. Although the problem of grouping and segmentation has a long history in computer vision (Wertheimer, 1923) and medical imaging (Yin *et al.*, 1994), it is still a significant challenge due to the specific nature of medical images. In the following section, we will discuss the binarization and thresholding, and regional-based segmentation methods, which are handy tools in digital image segmentation.

1.1.1 Binarization and Thresholding

Binarization is s simple but effective tool as a pre-processing step in several image processing applications, where gray levels of pixels belonging to the object, so-called foreground, are substantially different from the gray levels of pixels belonging to the background. Therefore, binarization algorithms divide the image into two classes of pixels, the background and the foreground pixels. The use of binary images decreases computational load for the subsequent overall applications. The applications of binarization include optical character recognition (OCR) for document analysis, scene matching, quality inspection of materials, and identifying the object of interest in medical imaging.

Classical image binarization techniques usually involve using a global or a local threshold to binarize the image. Thresholding technique is based on the assumption that pixels whose value lies within a certain range belong to the same class (Lim & Lee, 1990). Global thresholding algorithms use a single value over the entire image to binarize it into two classes, and local thresholding algorithms take into consideration the neighboring pixels when computing the threshold value. While thresholding is a widely used technique in image binarization, more advanced forms of thresholding can be used when an image foreground consists of more than one object and it is therefore necessary to segment it into more than two classes of pixels (Gonzalez & Woods, 2017).

Otsu global thresholding is among the first methods in thresholding. For an 8-bit grayscale image there are 256 different possible intensities, and so a grayscale level histogram graphically displays 256 numbers showing the distribution of the pixels amongst those grayscale values. Otsu's thresholding algorithm iterates through all possible thresholds from 0 to 255, i.e., maximum intensity value, to find a threshold which minimizes intra-class variance or maximizing inter-class variance. It aims to

find the minimum value of the weighted sum of the standard deviation for the two classes by

$$\sigma_{\omega}^2(t) = \omega_1\sigma_1^2 + \omega_2\sigma_2^2, \tag{1.1}$$

where the weights are the probability of classes and are computed from the normalized histogram (Otsu, 1979). Gajanayake *et al.* (2009) showed that the Otsu's thresholding method is the most suitable image segmentation method to segment a brain tumor from an MR Image. Saad *et al.* (2011) used histogram based thresholding techniques to detect brain lesions from Diffusion-weighted MRI. Phromsuwan *et al.* (2013) used Otsu's thresholding for binarization of magnetic nanoparticles.

The local thresholding algorithms by Niblack (1986) and Sauvola & Pietikainen (2000) have been used in many image binarization applications. In these algorithms, the choice for local threshold value is based on calculating the average and dispersion in neighborhood windows of each pixel. The threshold value is calculated by these methods using a local window. In Niblack algorithm, the estimation of the threshold in a local window defined by

$$T_{Niblack} = \mu + k\sigma, \tag{1.2}$$

where μ and σ are the mean and standard deviation of pixel intensity values in local window and k is a fixed value, usually chosen (empirically) as 0.2 to balance signal with noise. Sauvola improves the Niblack method by using the dynamic range of standard deviation R when computing the threshold by

$$T_{Sauvola} = \mu[1 + k(\frac{\sigma}{R} - 1)], \tag{1.3}$$

where R has the effect of amplifying the contribution of standard deviation in an adaptive manner. Consider, for example, a dark text on light dirty-looking back-ground. Senthilkumaran & Vaithegi (2016) proposed an efficient implementation for threshoding of Niblack and sauvola local thresholding algorithms and compared them using medical images.

Voting-based image binarization algorithms try to combine the two thresholding approaches by using them in image areas where they work the best. The most basic method of voting based image binarization presented in Pruncu *et al.* (2018) is the one which democratically chooses the value of a specific pixel, based on the results of four thresholding algorithms. If most algorithms decided that a pixel should be a foreground, then the output image will contain this pixel with value 0 or value 1 otherwise. This method can be further developed by adding weights to specific algorithms before probing the pixel values. This way, the solution will have a more personalized result (Pobleanu *et al.*, 2020). Kovacevic *et al.* (2002) a weighted based thresholding using local and global thresholding algorithms for automatic segmentation of brain images. Boiangiu *et al.* (2016) presented a voting-based algorithm to image binarization as a solution of Optical Character Recognition (OCR).

While global thresholding algorithms are high speed in comparison to local ones, they only perform well in cases where the input image histogram has a bimodal distribution, i.e., when the histogram contains two local maximum points (modes), corresponding to either object or background, consider an image with a good separation between background and foreground, or with a uniform lighting or contrast. Local algorithms overcome the problems presented by global methods by calculating pixel threshold values by probing information from neighboring pixels (Pobleanu *et al.*, 2020), therefore, they provide better results for images with uneven

lighting, but they are less resistant to the noise. In addition, local algorithms need to determine the size of the local window manually for each image under consideration to get better binarization result. A method for binarization of degraded documents proposed by Kaur *et al.* (2020) which modifies the Sauvola algorithm by automatically computing the window size dynamically across the image pixel to pixel. This method calculates the windows size for the pixel by using the stroke width of the pixel. The stork width of a pixel is the width of the most likely stroke containing the pixel. Since the text maintains a nearly constant or low variant stroke width over the entire image, this method is only useful for extracting the text from an image.

A more improved binarization technique is proposed by Costin-Anton *et al.* (2009), where foreground reconstruction-based binarization algorithm converts a grayscale document to black and white. This method can also be adapted to color images. This method reconstructs the foreground of the document by scanning the surrounding of a source point until a border between the foreground and the background is encountered. A source point is a pixel in the image surrounded in all directions by pixels having higher or equal values. More precisely, a source point represents a local minimum from the point of view of the grayscale pixel value.

Shaikh *et al.* (2013) proposed an iterative partitioning for binarization. This method starts by determining the peaks of the grayscale histogram of the entire image. If there are only two peaks, the binarization is trivial because one of the pixel groups represent the object while the other one represents the background. If there are more than two peaks, the image is divided into four rectangles and the grayscale peaks of each one are determined. The process continues until either the partition size has reached a pre-defined minimum or each partition has two or fewer sharp peaks.

1.1.2 Regional-Based Segmentation

The regional-based segmentation method in digital image processing is basically partitioning region R into sub-regions R_i, which are similar based on a set of logical predicates (Gonzalez & Woods, 2017). Gupta (2018) and Kang *et al.* (2009) classified and compare the image segmentation algorithms. Zhang *et al.* (2008) presented an evaluation survey on unsupervised image segmentation techniques. Shankar & Shyry (2021) and Tripathi *et al.* (2012) reviewed different image segmentation techniques for medical images. Some modern techniques of regional-based segmentation are graph-cut (Hochbaum, 2009) and adaptive contour segmentation (Feng *et al.*, 2013), which are more sophisticated and need high computational cost. The three main methods for regional-based segmentation which are still being used by many researchers are region-growth and splitting and merging segmentation. Also, some methods use a majority voting system for image segmentation, Boiangiu & Ioanitescu (2013) proposed a voting method that tries to merge different results of some well-known image segmentation algorithms including histogram based, graph based and region growing based segmentation.

Region-growth method starts with a seed cell or pixel, then the similar pixels will be appended to the seed pixel and grouped into the larger regions based on predefined criteria of growth. The seed cell usually depends on the nature of the problem. The challenges with region growing method are the rule for stopping the growth, and finding the seed cell when there is no prior information. However, the advantage of this method is that there is no need of predefined numbers of clusters. Fan *et al.* (2005) proposed a comparison study on seeded region growing algorithms. Preetha *et al.* (2012), Zhu *et al.* (2013) and Kang *et al.* (2012) modified the conventional region growing method to segment the image. Several studies including Hore *et al.* (2016), (Xu *et al.*, 2011) and (Fan *et al.*, 2001) have developed

integrated methods with region growing for image segmentation.

Saad *et al.* (2010) used the split and merge approach to segment brain lesions. Histogram thresholding technique is then used at each split level to detect pixels with either hyperintense or hypointense. Several statistical features are discussed and evaluated to select the best feature as homogeneity criteria. Hyperintense and hypointense lesions are segmented automatically by merging the regions that are homogenous according to the criteria.

1.2 Statement and Structure of Present Work

As described in greater details in the next chapter, the problem addressed in this dissertation concerns a system in which equal particles are suspended in an upward fluid flow. In the course of the evolution of the system occasionally clusters of particles form characterized by a faster or slower downward velocity. An analogy can be established between the particle velocity and the pixels of an image, and in this sense, our work shares similarities with image segmentation. However, there are also considerable differences because the particles forming the clusters are not necessarily very close to each other, but are spread over regions that contain also particles with velocities closer to the mean. This fact has required several modifications to the techniques developed in the image processing field.

The structure of this study is as follows: In Chapter 2, we describe the data that we used in our study and how they are obtained. In Chapter 3, we describe different attempts that we have made to identify anomalous particles and particle groups, and we conclude with the description of the method used to find a common attribute to use for clustering. In Chapter 4 we illustrate the random masks method that we use as a pre-step to binarizing the region. Then, we illustrate how we use thresholding to binarize the image based on 'hits' of each cell. At the end of the

chapter, we discuss how we identify the clusters over the entire region and how we track them over time. In Chapter 5, we show the performance of the method that we have developed by applying it to the data described in Chapter 2. In Chapter 6 describes the conclusions of this dissertation and indicates possible directions for future work.

Chapter 2

Data

In this chapter we describe how the data used to verify our methodology is obtained, then we describe the different attributes of the data and how they are correlated. At the end of section, we explain the reason why we choose the location and the velocity of the particles as the basis for the clustering.

2.1 Physalis Method

The data of our study is obtained from a resolved simulation framework called "Physalis" method which simulates equal particles suspended in an upward fluid flow. In this section, we briefly describe the method. The reader is referred to the more recent studies by Sierakowski & Prosperetti (2016) and Willen *et al.* (2017) for a more complete description of the method and details of its implementation.

The Physalis method, focuses on one category of multiphase flow, namely particulate flows, in which the dispersed phase consists of solid spherical particles. A typical example is a fluidized bed, where initially stationary particles become suspended in an upward-directed fluid stream once the flow velocity exceeds the so-called minimum fluidization velocity.

The flow of an incompressible fluid is governed by the Navier-Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{g}, \tag{2.1a}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1b}$$

where $\mathbf{u}$ is the fluid velocity, p is the pressure, $\mathbf{g}$ the gravity force per unit mass, ρ and ν are the fluid density and kinematic viscosity. These equations are subject to no-slip and no-penetration boundary conditions at the surfaces of all the particles, which means that

$$\mathbf{u} = \mathbf{w}_p + \mathbf{\Omega}_p \times (\mathbf{x} - \mathbf{x}_p), \quad \text{on} \quad S_p. \tag{2.2}$$

Here $\mathbf{w}_p$ is the velocity of the particles, $\mathbf{\Omega}_p$ the angular velocity, $\mathbf{x}_p$ the position of the center, and S_p the surface of the particles. The linear momentum equation for each particle is

$$\rho_p v_p \frac{d\mathbf{w}_p}{dt} = \oint_{S_p} dS \boldsymbol{\sigma} \cdot \mathbf{n} + \sum_j \mathbf{F}_j + \rho_p v \mathbf{g}. \tag{2.3}$$

Here ρ_p and v_p are particle density and volume, $\mathbf{F}_j$ is the direct force force, e.g. due to collision, from the j^{th} particle to the particle considered. The first term in the right hand side represents the hydrodynamic force the fluid exerts on the particle, $\mathbf{n}$ is unit the normal pointing outward of the particle surface S_p, and $\boldsymbol{\sigma}$ is the fluid stress tensor

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T), \tag{2.4}$$

where $\mu = \rho\nu$ is the dynamic viscosity of the fluid, and $\mathbf{I}$ is the second order identity tensor. Similarly, the angular momentum balance for each particle can be written as

$$I_p \frac{d\mathbf{\Omega}_p}{dt} = \oint_{S_p} dSx \times (\sigma \cdot \mathbf{n}) + \sum_j \mathbf{L}_j, \tag{2.5}$$

where I_p is the moment of inertia of the particle, and $\mathbf{L}$ is inter-particle torque due to collisions. Collisions are described by a Hertzian model which includes a coefficient of restitution and elastic properties of the particles.

Equations 2.1, 2.2, 2.3 and 2.5, together with proper initial conditions and external boundary conditions, form an initial-boundary-value problem, which typically can only be solved numerically. The particles represent internal boundaries for the fluid and therefore, they are intimately connected to the flow. In order to quantify this connection, the Physalis method uses an analytical solution valid near the particle surface as a 'bridge' from the particle surface to the nearest nodes of the fixed Cartesian grid on which the Navier-Stokes equations are integrated. The validity of the analytical solution rests on the no-slip condition, which requires the relative velocity between the fluid and the particle in the neighborhood of the particle surface to be very small. This small relative velocity permits to linearize the Navier–Stokes equations around the state of zero motion in the rest frame of the particle reducing them, in effect, to the Stokes equations, which are

$$\nabla \cdot \tilde{\mathbf{u}} = 0, \tag{2.6a}$$

$$-\nabla \tilde{p} + \mu \nabla^2 \tilde{\mathbf{u}} = 0. \tag{2.6b}$$

Here $\tilde{\mathbf{u}}$ is the velocity field in the particle rest frame and $\tilde{p}$ the corresponding pressure field which includes contributions arising from the change of reference frame from the laboratory frame to the particle rest frame. The solution of the above equations 2.6

over a stationary sphere, expressed in the spherical coordinate system whose origin is the particle center, $\mathbf{x} = (r, \theta, \phi)$, can be synthetically written as

$$\tilde{p}(\mathbf{x}) = \frac{\mu\nu}{a^2} \sum_{n=0}^{\infty} \mathcal{L}_1(p_n, \phi_n, \chi_n, \mathbf{x}, n), \tag{2.7}$$

$$\tilde{\mathbf{u}}(\mathbf{x}) = \frac{\nu}{a} \sum_{n=0}^{\infty} \mathcal{L}_2(p_n, \phi_n, \chi_n, \mathbf{x}, n), \tag{2.8}$$

where a is the particle radius, p_n, ϕ_n, χ_n are solid harmonics of order n, and $\mathcal{L}_1$, $\mathcal{L}_2$ are operators that apply spatial derivatives to p_n, ϕ_n, χ_n. The explicit form of them can be found in Sierakowski & Prosperetti (2016). p_n, ϕ_n, χ_n have the following form

$$p_n(\mathbf{x}) = (\frac{r}{a})^n \sum_{-n}^{n} p_{nm} Y_n^m(\theta, \phi), \tag{2.9}$$

$$\phi_n(\mathbf{x}) = (\frac{r}{a})^n \sum_{-n}^{n} \phi_{nm} Y_n^m(\theta, \phi), \tag{2.10}$$

$$\text{and} \quad \chi_n(\mathbf{x}) = (\frac{r}{a})^n \sum_{-n}^{n} \chi_{nm} Y_n^m(\theta, \phi), \tag{2.11}$$

where Y_n^m is a spherical harmonic of order n and degree m, and p_{nm}, ϕ_{nm} and χ_{nm} are constants referred to as Lamb coefficients. The details of the Lamb coefficients determination is discussed in Sierakowski & Prosperetti (2016). An advantage of the Physalis method is that the hydrodynamic force, couple, stresslet, etc. of each particle are directly obtained from the Lamb coefficients which are a byproduct of the solution method itself. In particular the stresslet, which is the contribution of the

particles to the stress in the fluid particle mixture, is defined by

$$s_{ij}^{p} = \frac{1}{2}\oint_{S_p} [(\sigma \cdot \mathbf{n})_i x_j + (\sigma \cdot \mathbf{n})_j x_i - \frac{2}{3}(\mathbf{x} \cdot \sigma \cdot \mathbf{n})\delta_{ij}] ds. \tag{2.12}$$

2.2 Simulations and Data

In this section, we briefly describe the simulations performed with the Physalis method to obtain the data used in our study. These simulations were conducted in a computational domain R in the shape of a parallelepiped with a square cross section and aspect ratio (vertical/horizontal) = 3. The height of the domain was $60a$ while the horizontal size was $20a \times 20a$, where a is the equal radius of all the particles. The boundary conditions on all six boundaries were periodicity conditions. The total number of cells used in the simulation was $480 \times 160 \times 160$ with 8 cells per particle radius. A downward pressure gradient was imposed to balance the gravitational forcing in the vertical direction. Due to this arrangement, the mean settling velocity of the spheres vanishes. Among the data generated in previous studies of resolved spherical particles simulations presented in Willen *et al.* (2017), we consider here the data of the simulations with 1000 particles having a density ratio to the fluid given by $\rho_p/\rho_f = 2$ occupying a volume fraction of 17.4%. The particle volume fraction is defined as the total volume of particles over the total volume of the box

$$\phi = \frac{N_p(\frac{4}{3}\pi a^3)}{60a \times 20a \times 20a}, \tag{2.13}$$

where N_p is the total number of particles, and $20a \times 20a \times 20a$ is the total volume of the region R. The ratio of inertial to viscus forces is characterized by the Galilei

Table 2.1: Simulation parameter values

parameters	meaning	value
N_p	number of particles	1000
ρ_p/ρ_f	density ratio	2
E (MPa)	Young's modulus	0.65
σ	Poisson ratio	0.5
e_{dry}	dry coefficient of restitution	0.98
Re_t	single-particle Reynolds number	43.27
Ga	Galilei number	49.7
$a/\Delta x$	cells per particle radius	8
X_s/a, X_e/a	limits of computational domain in x-direction	-21, 21
X_n	number of grids in x-direction	160
Y_s/a, Y_e/a	limits of computational domain in y-direction	-21, 21
Y_n	number of grids in y-direction	160
Z_s/a, Z_e/a	limits of computational domain in z-direction	-63, 63
Z_n	number of grids in z-direction	480
$\nu t/(2a)^2$	duration of the simulation	24.3

number defined by

$$Ga = \frac{1}{\nu}\sqrt{\left(\frac{\rho_p}{\rho_f} - 1\right) g(2a)^3}. \tag{2.14}$$

The Reynolds number based on the terminal fall velocity w_t of an isolated particle is

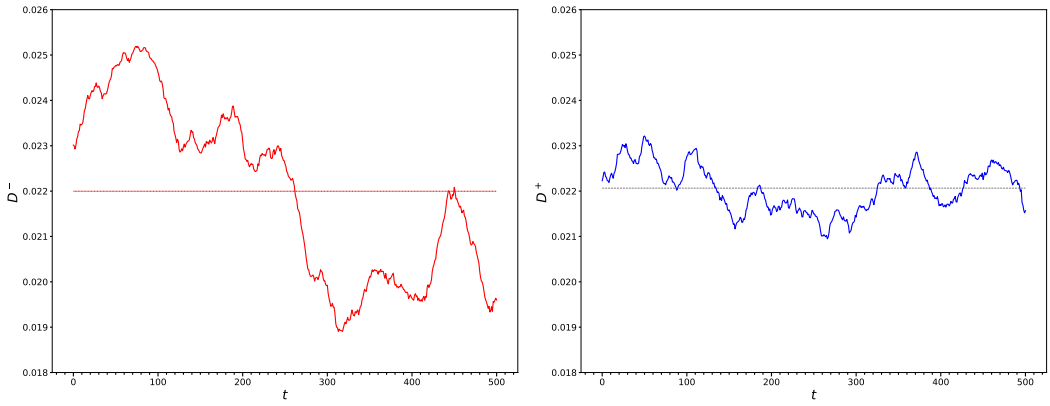
$$Re_t = \frac{2aw_t}{\nu}. \tag{2.15}$$

Table 2.1 summarize the values of parameters used as the input for both particle and fluid model to obtain our data.

Initially the particles are randomly arranged in the computational domain and, before data were recorded, allowed to reach a statistically steady state as revealed by the average values of the fluid velocity and the pressure drop. After reaching steady state, at each time step, all the particle attributes such as position, orientation, velocity, etc. (see list in table 2.3) are recorded.

Table 2.3: The attributes of our data and the description

attribute meaning	name
coordinate of the particle in x-direction	Coordinatex
coordinate of the particle in y-direction	CoordinateY
coordinate of the particle in z-direction	CoordinateZ
particle velocity in x-direction	VelocityX
particle velocity in y-direction	VelocityY
particle velocity in z-direction	VelocityZ
particle acceleration in x-direction	AccelerationX
particle acceleration in y-direction	AccelerationY
particle acceleration in z-direction	AccelerationZ
angular velocity of particle in x-direction	AngularVelocityX
the hydrodynamic force exerted on the particle in x-direction	HydroForceX
the hydrodynamic force exerted on the particle in y-direction	HydroForceY
the hydrodynamic force exerted on the particle in z-direction	HydroForceZ
collision and lubrication forces in x-direction	InteractionForceX
collision and lubrication forces in y-direction	InteractionForceY
collision and lubrication forces in z-direction	InteractionForceZ
stress tensor in xx-plane	S_{xx}
stress tensor in yy-plane	S_{yy}
stress tensor in zz-plane	S_{zz}
stress tensor in xy-plane	S_{xy}
stress tensor in xz-plane	S_{xz}
stress tensor in yz-plane	S_{yz}



(a) (b)

S

Figure 2.1: Time dependence of the fluctuations of the standard deviation of the particle velocity in the vertical directions with respect to the time-mean values indicated by the horizontal lines for a) falling particles and b) rising particles

2.3 Vertical Velocity Description

For the reasons explained in the next chapter, for the purposes of this study, the data that we have found most useful concerned the vertical velocity of the particles.

Figures 5.20a and 5.20a show the time dependence of the standard deviations of the particle velocity split into falling and rising particles, D^+ and D^-, respectively. We can see from these figures that the standard deviation of the falling particles fluctuates more than that of the rising particles.

Figure 2.2 shows the mean upward and downward velocities of all the particles vs time and compares it with the overall mean velocity which is very close to zero as expected (recall that the purpose of the applied upward pressure gradient is indeed to induce a fluid velocity able to suspend the particles on average). It may be noted that the standard deviation of the vertical velocities is around 0.02 while the means are around ± 0.03. This implies a strongly fluctuating system.

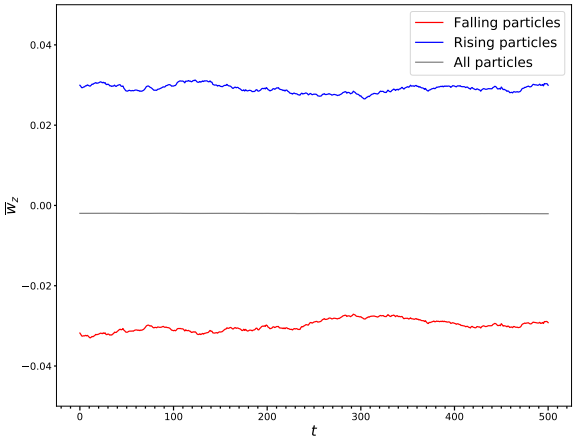


Figure 2.2: The average particle velocity in the vertical directions for rising particles (blue line), all particles (gray line) and falling particles (red line) vs time

Chapter 3

Feature Selection

In the initial phase of this study, we explored whether there was significant correlation among different attributes of individual particles, or particles existed that were outlier compared with the remaining ones. For this purposes, we used recent methods developed in data science.

3.1 Pearson Correlation

Figure 3.1 shows an example of the Pearson correlation coefficient among all the single particle attributes for a single time step. As can be seen in the figure, this analysis did not lead to particularly interesting results. For example, one can notice that the particle acceleration is correlated with the particle interaction force, which is not surprising because the interaction force is a major contributor to the total force acting on each particle. Another strong correlation is observed among the three diagonal components of the particle stresslet. This quantity represents the contribution of the particles to the effective viscosity of the mixture and, since there is no preferred direction in the motion of the individual particles, it is not surprising that all three directions give a similar contribution which manifests itself in an apparent

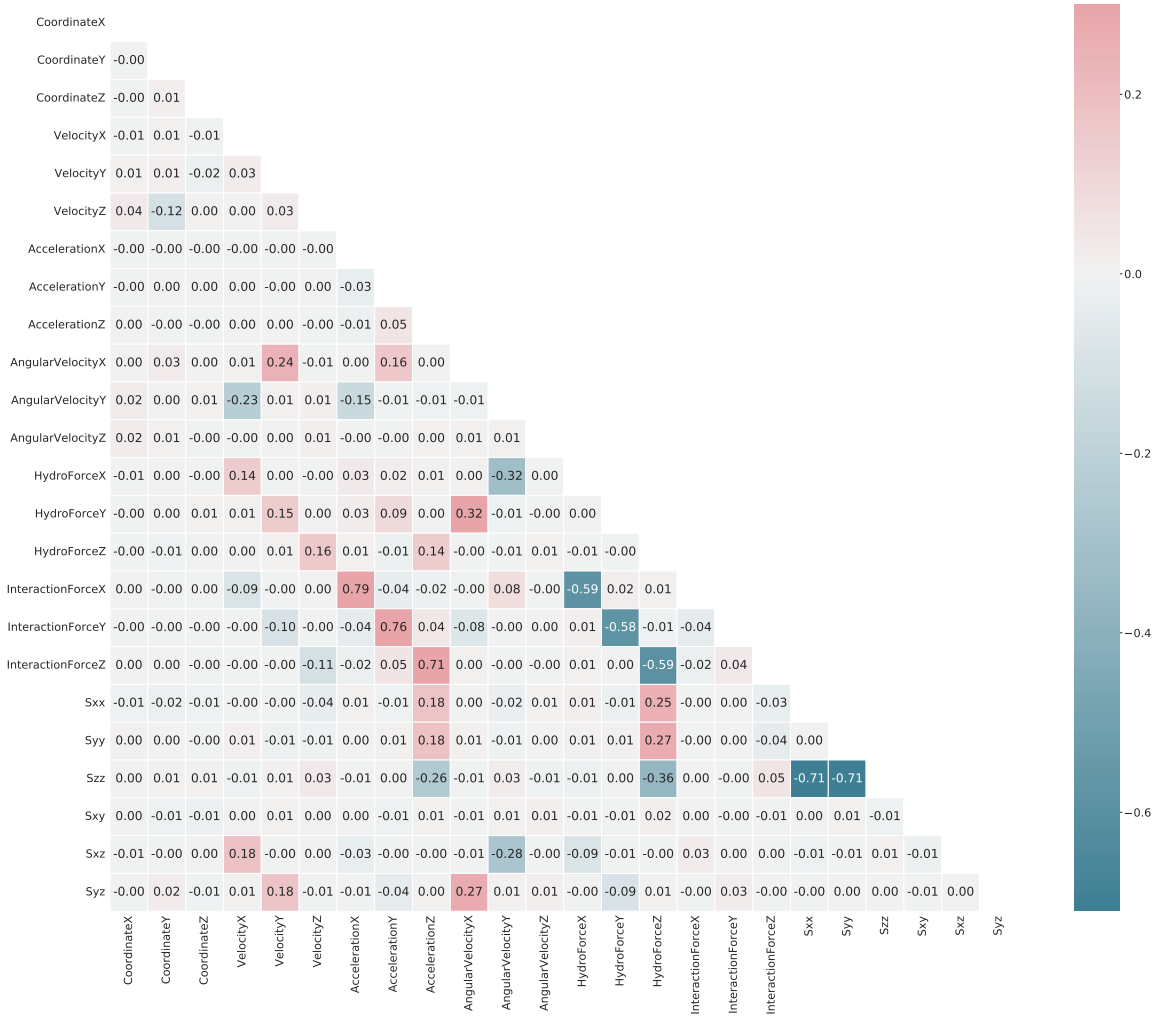


Figure 3.1: The correlation coefficients of all attributes in the data

correlation. Thus, none of these correlations is particularly interesting or unexpected. Several other attributes have a correlation index in the range around 0.5 or smaller which is not indicative of a significantly strong correlation.

The table shown in the figure is just one example of many others for different time steps that we have considered. However, it is quite typical and this type of analysis leads to the conclusion that there are no interesting correlations among the different attributes of the same particle.

3.2 Anomaly Detection

The next type of analysis that we embarked on was an attempt to identify outlier particles, i.e., particles possessing values of some attributes that distinguished them from the others. There are several different algorithms to pursue this objective, which are conveniently combined in the Python toolkit called "PyOD" (Zhao *et al.*, 2019). The reader is referred to PyOD Github for the details of implementation.

Among the methods available, we used Angle-based Outlier Detector (ABOD), Cluster-based Local Outlier Factor (CBLOF), Feature Bagging, Histogram-base Outlier Detection (HBOS), Isolation Forest, K Nearest Neighbors (KNN), Average KNN, Local Outlier Factor (LOF), Minimum Covariance Determinant (MCD), One-class SVM (OCSVM), Principal Component Analysis (PCA), Locally Selective Combination (LSCP).

An outlier is a point that is distant from other points, so the outlier score is defined by distance. The decision function in PyOD calculates the distance or the anomaly score for each data point. For example, the K Nearest Neighbors uses the distance and Average KNN uses the average distance to the k^{th} nearest neighbor as the outlier score. Another example is Principal Component Analysis, given the sum of weighted projected distances to the eigenvector hyperplanes, the

data with the larger distance, i.e., larger anomaly score, are assigned to the abnormal cluster. Thus, by finding the anomaly score the dataset will be binarized into two clusters abnormal data, so-called outliers and normal data. In our study we use terms interesting and non-interesting. Zhao *et al.* (2019) used a benchmark dataset for selected algorithms to provide an overview of the implemented models. The dataset has 200 samples randomly uniform generated with 25% of samples generated in which they are abnormal and labeled as 0 for groundtruth, and the rest of samples are labeled as 1. Figure 3.2 compares several detection algorithms by plotting decision boundaries and the number of decision boundaries. An outlier fraction is defined by the user to calculate the decision scores in order to identify entities possessing the attributes of interest. The same decision function used to predict the decision scores is used on the attributes to find the decision boundaries, i.e., the red dashed lined contours in the figure.

We have applied the algorithms contained in the PyOD toolkit on our data for different combinations of attributes and outlier fractions for several time steps. Figure 3.3 shows a case in which the outlier fraction is taken to be 10%. In this example, we used the particle physical coordinates, velocity components and the magnitude of the angular velocity and interaction force as an attribute subset at a single time step. As we can see from this figure the outlier detection algorithms do not identify abnormal data which prevents them from clustering our dataset.

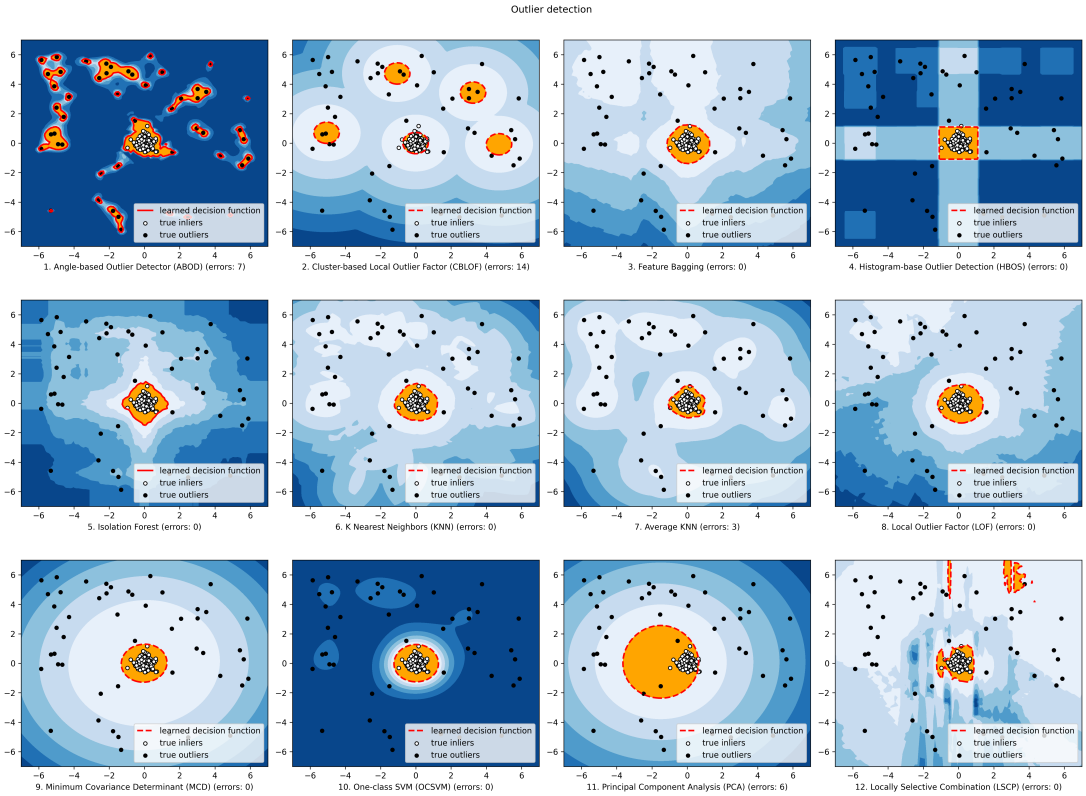


Figure 3.2: Comparison of several distance-based anomaly detection algorithms for a benchmark dataset

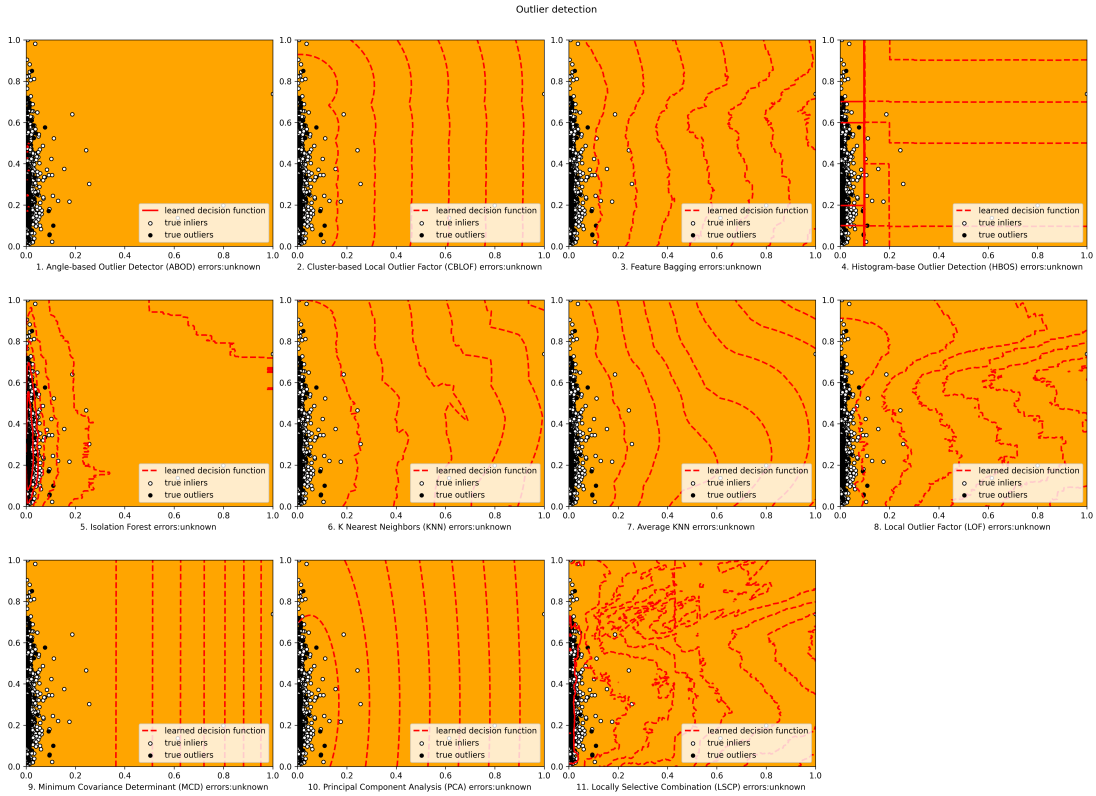


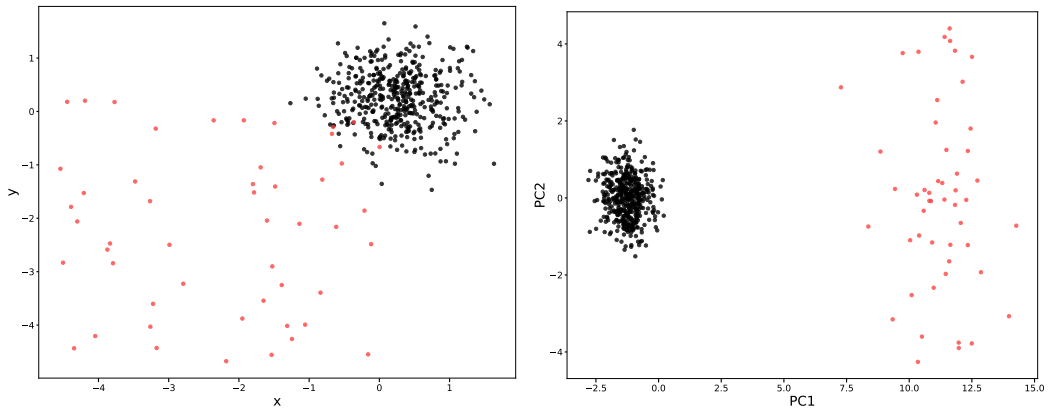
Figure 3.3: Distance-based anomaly detection algorithms implemented on our data

Many distance-based techniques, e.g. KNNs, used for anomaly detection detecting suffer the curse of dimensionality when they compute distances of every data point in the full feature space. High dimensionality has to be reduced. Interestingly, during the process of dimensionality reduction outliers are identified. Anomaly detection can be a by-product of dimension reduction. Therefore, we used AutoEncoder algorithm as another technique for anomaly detection.

Recall that the Principal Component Analysis uses linear algebra to transform the feature space. In contrast, the AutoEncoder technique can perform non-linear transformations with their non-linear activation function and multiple layers. It is more efficient to train several layers with an AutoEncoder, rather than training one huge transformation with Principal Component Analysis. The AutoEncoder techniques thus are used when the data problems are complex and non-linear in nature.

We have used PyOD toolkit to implement AutoEncoder algorithm on a benchmark dataset and our data. The benchmark dataset includes 1000 samples with 25 attributes in which 10% of the data are being contaminated. Figure 3.4a, on the left side, shows two attributes of benchmark dataset plotted against each other and figure 3.4b, on the right side, shows these attributes transformed on using two first principal components. The right figure shows that the outliers have been detected. The model has five layers. The input layer and the output layer has 25 neurons each, and there are three hidden layers with 10, 2, and 10 neurons respectively. We used a 70%-30% training and testing split on our dataset. Figure 3.5 shows the histogram of anomaly scores applying the above-mentioned AutoEncoder model on the benchmark data. We then set a threshold of 10 to binarize the data using the anomaly scores. We examined that the two clusters show a significantly different mean value for all the attributes of this benchmark dataset.

Figure 3.6 represents the anomaly scores applying the same algorithm on our data. As we can see from the figure, setting a threshold of 7, the model has been only able to detect 0.4% of particles as outliers. Therefore, we conclude that the AutoEncoder is not able to binarize our dataset on all the features space.



(a) (b)

Figure 3.4: a) two attributes of a benchmark dataset including 1000 random samples with 10% outliers b) attributes being transformed using Principal Component Analysis

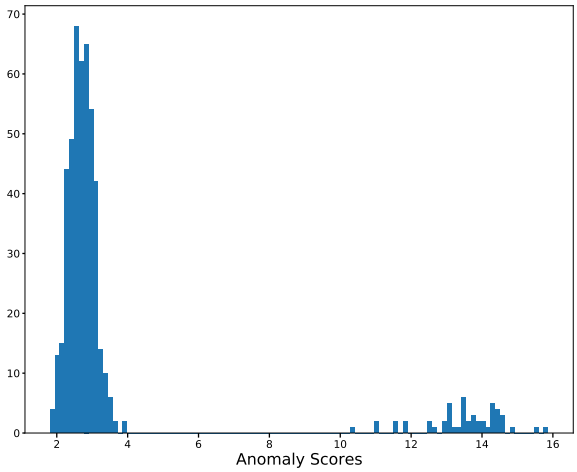


Figure 3.5: The anomaly scores of the AutoEncoder model implemented on benchmark dataset

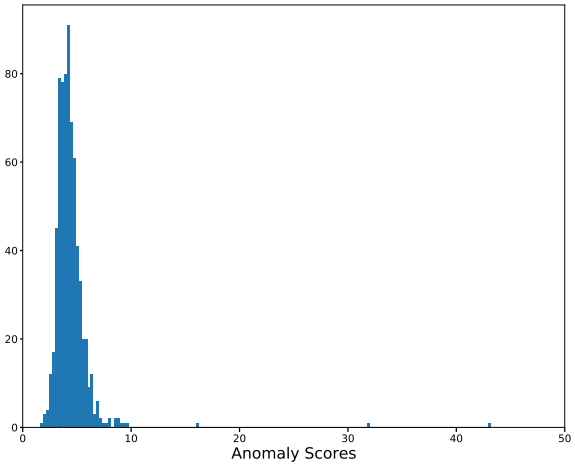
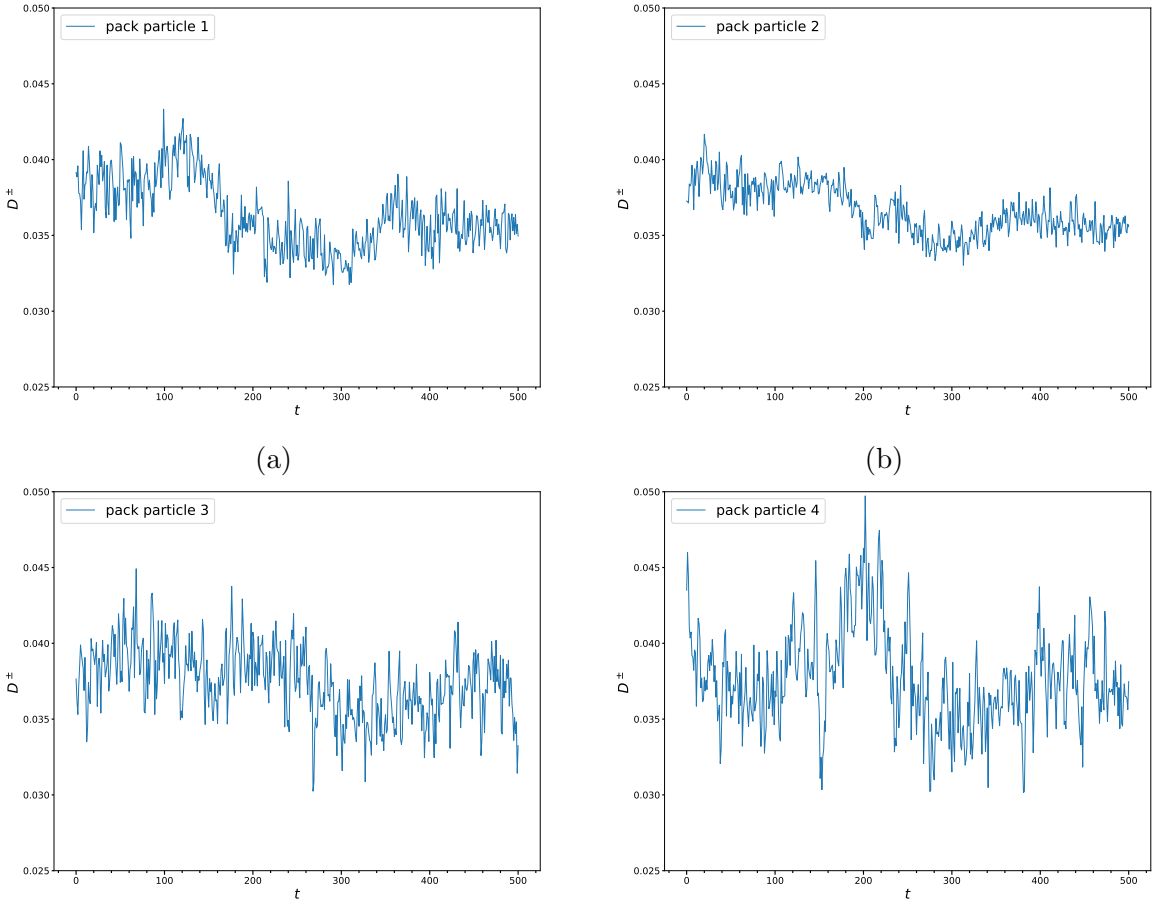


Figure 3.6: The anomaly scores of the AutoEncoder model implemented on our data

3.3 Voronoi Volume

We have implemented Voro++ library (Rycroft, 2009) to calculate the three dimensional Voronoi cells, i.e., Voronoi Volumes, in physical, velocity, hydrodynamic and interaction force spaces. We have used four bins to classify the Voronoi volumes in the physical space. The bins are defined as 0 to 6, 6 to 9, 9 to 12 and 12 to 20. Each bin results in a pack of particles which is about 59%, 14%, 9% and 18% of all particles respectively. We compared the standard deviation of vertical velocity over all time steps for each pack in figure 3.7 and the average of vertical velocity in 3.8.

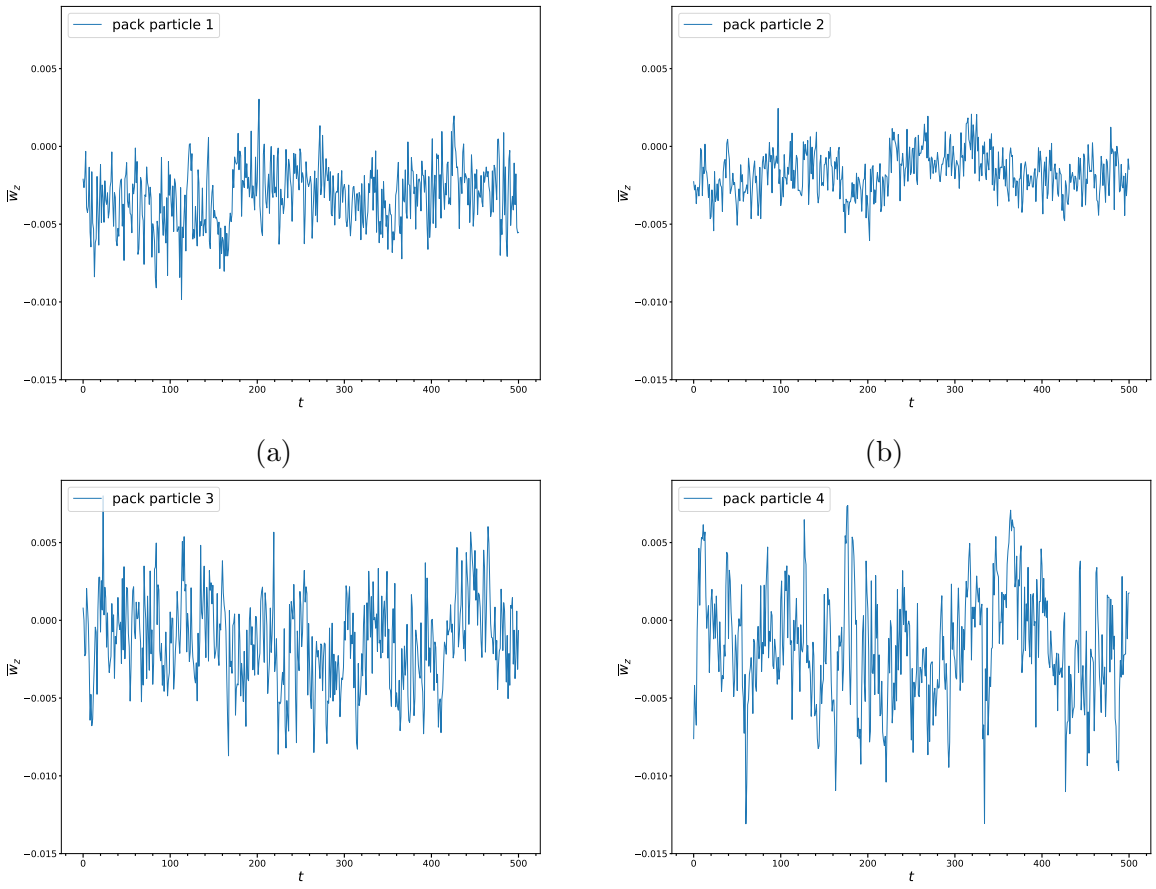
We cannot observe any specific pattern or correlation between the standard deviation of vertical velocity and the pack of particles nor the average of vertical velocity and pack of particles. In addition, the first pack are the particles with the smallest Voronoi volume, which means these particles are closer to the other particles. Since the ratio of this pack averaged over time is 59%, we can conclude that this pack can not be cluster of particles grouped together.



(a) (b)

(c) (d)

Figure 3.7: The standard deviation of vertical velocity for four pack of particles classified on the basis of Voronoi volumes of physical space vs time



(a) (b)

(c) (d)

Figure 3.8: The average of vertical velocity for four pack of particles classified on the basis of Voronoi volumes of physical space vs time

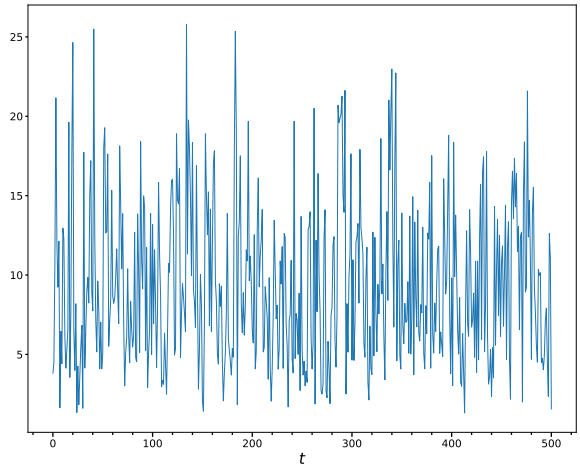


Figure 3.9: The distance of centroids of two subcategories on the basis of Voronoi volumes in velocity space vs time

We also classified the Voronoi volume quantity of velocity space into two subcategory, the particles the Voronoi volume of velocity less than 0.0015 and bigger than 0.0015. Majority of particles on average, i.e., 98.5%, are assigned to the second category. Therefore, we can not conclude this category is identified as a cluster. We have also found the distance of centroid of these categories vs time (see figure 3.9). We can see from this figure that this distance fluctuates significantly over time that can not lead us to any conclusion. The distance of these centroid at each time step from the next time step is also depicted in figure 3.10.

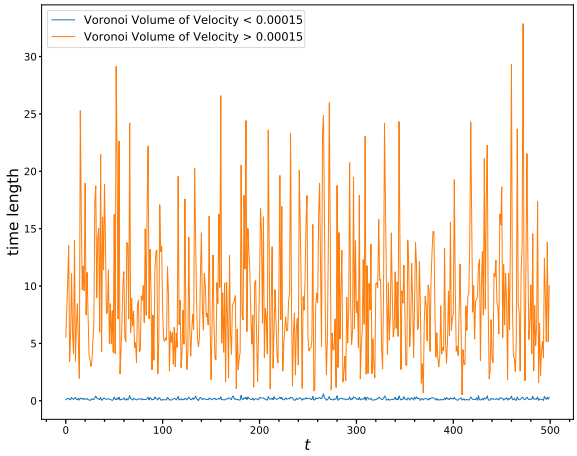


Figure 3.10: The distance of centroids of two subcategories on the basis of Voronoi volumes in velocity space from the next time step

Another investigation on the Voronoi volume is finding the correlation between standard deviation of all attributes in all time steps, standard deviation of Voronoi volume in velocity and physical space and the maximum of probability density of the Voronoi volume in physical space in each time step. Figure 3.11 shows this correlations using Pearson coefficient. From this figure the only significant correlation is between standard deviation of the velocity in z- and y- direction with standard deviation of Voronoi volumes in velocity space. This means when, for example, the standard deviation of vertical velocities is larger, the variation in velocity is larger and standard deviation of the Voronoi volume would be larger. This is an obvious physical fact which does not lead us to any potential clusters for the particles. We could say the Voronoi volume is a potential means to cluster the particles only if there was a high correlation between standard deviation of the Voronoi volume in physical space and the standard deviation of vertical velocity.

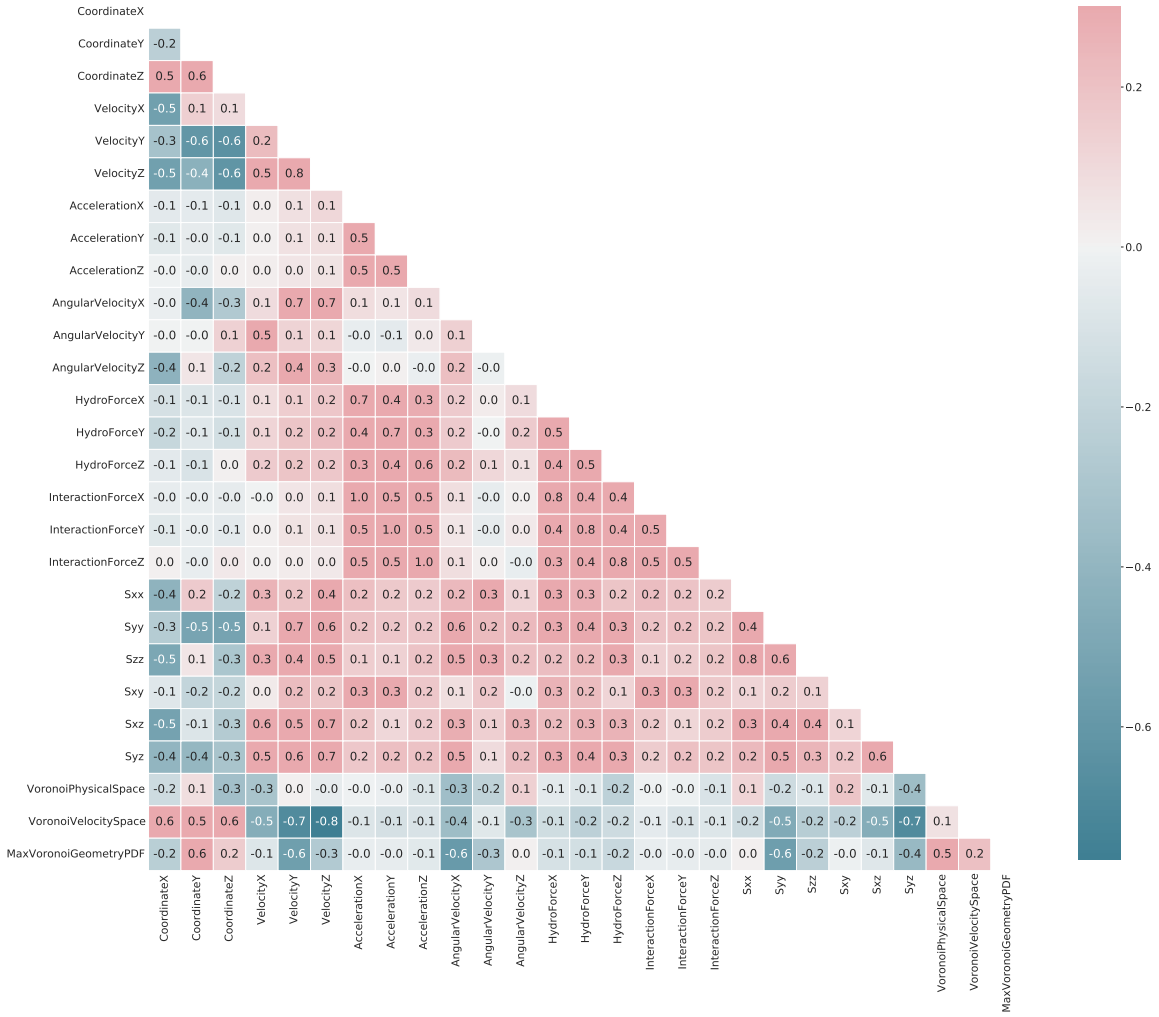


Figure 3.11: The correlation coefficients of standard deviation of all attributes and Voronoi volumes quantities over time

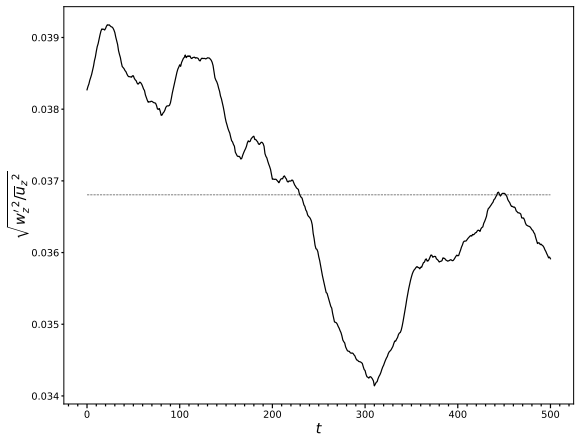


Figure 3.12: Time dependence of the fluctuations of the normalized standard deviation of particle vertical velocity with respect to the time-mean values indicated by the horizontal lines for all particles

3.4 Vertical Velocity Selection

We showed in a previous section of this chapter that there are no meaningful correlations between single attributes of individual particles. We then tried to identify outlier particles by distanced-based methods including different subsets of attributes, but, we concluded that the data are too random to be cleanly divided into inlier and outlier groups. We then tried a dimensionality reduction method in the hope of identifying particularly significant attributes able to differentiate the particles, but, this attempt also failed. Finally, turning to Voronoi volumes analysis, we studied whether spatial proximity (in physical or other spaces) promoted similarities among the attributes of proximal particles, but again, the answer was negative.

These negative results lead us work therefore to investigate whether the particles form clusters the members of which share anomalous values of some attributes. The starting point of this idea is similar to that prompted the Voronoi volumes study. However, that study considered the entire computational domain R, and therefore, could not reveal the presence of localized clusters. We have therefore developed a method suitable for this purpose which is described in the next chapter.

We examine whether particles form spatial clusters the members of which are animated by larger than average velocity components in the vertical direction. This study of this particular attribute was prompted by results such as those of figure 3.12 which shows the normalized standard deviation of the particle vertical velocity over time defined by

$$\sqrt{\frac{\langle [w_z'(t)]^2 \rangle}{\langle u_z^2 \rangle}} = \sqrt{\frac{\langle [w_z(t) - \langle w_z \rangle]^2 \rangle}{\langle u_z^2 \rangle}}, \tag{3.1}$$

where $\langle w_z \rangle$ is the vertical particle velocity averaged over time and over all the particles, $\langle [w_z'(t)]^2 \rangle$ is the time dependent velocity fluctuations averaged over all the particles and $\langle u_z^2 \rangle$ is the mean square vertical fluid velocity. The figure shows that there are long intervals of time during which the standard deviation of the particle vertical velocity is significantly above or below the mean value indicated by the horizontal line. This fact suggest that, during these intervals of time, there may be relatively large numbers of particles with vertical velocities above or below the mean value.

Chapter 4

Methodology

As we explained in Chapter 2 and 3, the average vertical velocity of particles over the entire region is zero. We also showed that the instantaneous fluctuations of the mean square of the particle velocity during some time intervals significantly differ from its average value over time. This means that, in those intervals of time, there is a significant number of particles which have a vertical velocity with a modulus large enough to affect the mean square fluctuations. (Willen & Prosperetti, 2019) hypothesized that these particles tend to cluster together. The objective of our study is to find these clusters in the computational domain which is the region R of our interest.

In the present study the vertical velocity of the particles is the attribute that we are interested in. We describe a general method that can identify the spatial clusters of particles responsible for the observed anomalous fluctuations. The particles are contained in the region R used for the simulations described in Chapter 2. The actual computational domain is a three-dimensional parallelepiped but, for ease of exposition, we will consider a two-dimensional region. Extension of the method from two to three dimensions is immediate and will be described at the end of the Chapter. In the tables 4.1 and 4.3, we describe the subscripts and the variables used in this

Chapter.

Table 4.1: Nomenclatures of variables used in the current Chapter

variable	meaning
R	region
L	length of region R
l	length of random mask
N	number of cells
$N_{particles}$	number of particles
$\overline{N}$	number density of particles in the region R
ℓ	cell side
c	cell
C	cell array
i	x-coordinate of bottom-left corner of a cell
j	y-coordinate of bottom-left corner of a cell
k	z-coordinate of bottom-left corner of a cell
m	random mask
N_m	number of random masks
$f(m)$	logical predicate to find interesting mask
$\overline{W}_z^{\pm}$	average vertical velocity of rising or falling particles in the region R
w_z	the vertical velocity of rising or falling particles
$D^{\pm}$	the standard deviation of velocity of rising or falling particles region R
$N_m^{\pm}$	total number of rising or falling particles in each mask
$n_m^{\pm}$	total number of interesting rising or falling particles in each mask
$f_c(l)$	logical predicate to find interesting cells
h	counter value of a cell
T	global threshold for finding interesting cells
$\mathcal{C}$	cluster

Table 4.3: Subscripts and their meaning

subscripts	meaning
x	x-direction
y	y-direction
z	z-direction
l	cell label
m	random mask
J	cluster index
p	particle

4.1 Identifying Interesting Cells

The first step is to divide the region R into equal cells. The next step is to identify the interesting cells, i.e, the cells that contain particles with a vertical velocity anomalously large in modulus. We will proceed with 'random masks' and global thresholding to binarize the region R into interesting and non-interesting cells. After binarizing the region and identifying the interesting cells, we need to establish the links between these interesting cells to identify the spatial clusters. Thus, we will use a method which is similar to region growing method integrated with region merging to find these groups of connected cells and thereby clusters in the domain. The last step is to track the clusters in time. We will describe the details of our method in the following sections

In two dimensions, the region R is rectangular with area $L_x \times L_y$, in which L_x is the length of R in x-direction and L_y is the length of R in y-direction. In three dimensions there is a length L_z in the z-direction and a corresponding volume $L_x \times L_y \times L_z$. The region R contains $N_x \times N_y$ equal square cells with N_x the number of cells in x-direction, and N_y is the number of cells in y-direction. Correspondingly, in three dimensions the total cells are cubic with a total number $N_x \times N_y \times N_z$.

The average number density of particles in the region R is $\overline{n} = N_{particles}/L_x L_y$. Therefore, on average, a cell of side ℓ would be expected on average to contain $\ell^2 \times \overline{n}$ particles. We have chosen ℓ in such a way that there is one particle per cell on average.

It is convenient to place the origin of a Cartesian coordinate system at the bottom-left corner of the region R. In our two-dimensional explanation, the cells have equal sides which we redefine as the unit of length. Each cell is identified by the coordinates (i, j) of its bottom-left corner with i the coordinate in the x direction and j the coordinate in the y direction or, in three dimensions, by the triplet (i, j, k).

We also label the cells sequentially starting from the one with its bottom left corner at the origin which is given the label c_0, proceeding with the one above it, labeled c_1, then the next one above, labeled c_2 and so on to the last cell of the column at the top left, which is labeled c_{N_y-1}. The next cell is the one adjacent to the cell c_0 which is labeled c_{N_y}, then the one above, labeled c_{N_y+1} and so on. In this way each cell receives a label l according to the formula

$$l = j + i \times N_y \,. \tag{4.1}$$

Therefore, the cell at the bottom-right of the region R receives the label $c_{N_y \times (N_x-1)}$, and the cell in the top-right corner is labeled as $c_{N_y \times N_x - 1}$. In three dimensions the rule corresponding to 4.1 is

$$l = k + j \times N_z + i \times N_z \times N_y \,. \tag{4.2}$$

Figure 4.1 shows an example of a region R with $L_x = 10$ and $L_y = 15$ divided into 10×15 cells, in which $N_x = 10$ and $N_y = 15$. For instance, using equation 4.1 the cell with coordinates $(5, 10)$ is labeled as c_{85}. The small circles indicate the particles.

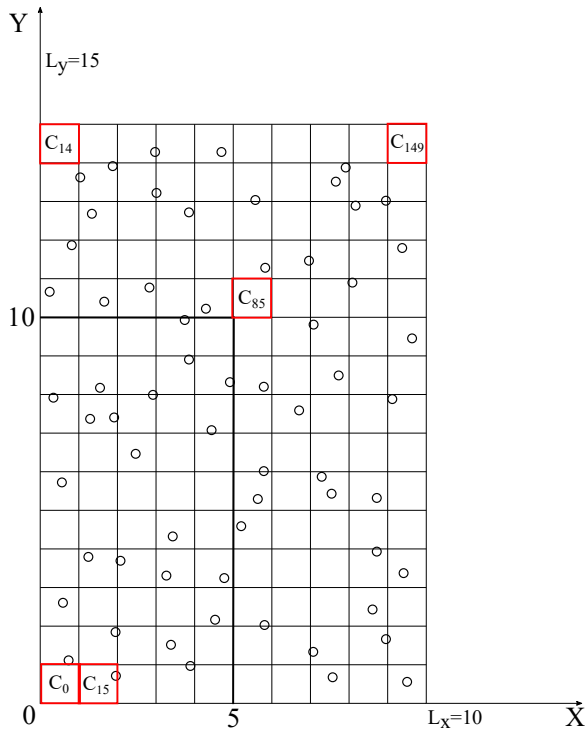


Figure 4.1: A region with $L_x = 10$ and $L_y = 15$ divided into 10×15 number of cells. The circles indicate the particles.

We need to store the information of each cell for further implementation. Therefore, we will store the cell labels and the bottom-left corner coordinates of each cell in array C

$$C = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 2 & 0 & 2 \\ \vdots & \vdots & \vdots \\ (N_x \times N_y) - 2 & L_x & L_y - 1 \\ (N_x \times N_y) - 1 & L_x & L_y \end{bmatrix}, \tag{4.3}$$

where the first column is the label assigned to each cell, and the second and third columns are the x- and y-coordinates of the bottom-left corner of the cell.

The next step is identifying the interesting cells, i.e., cells that contain particles with anomalously large (in modulus) velocities. Let $\overline{W}_z^+$ be the average velocity of rising particles in the entire region R and $\overline{W}_z^-$ the corresponding average velocity of falling particles. Let also D^+ be the standard deviation of the velocity of the rising particles in the region R and, D^- the corresponding standard deviation of the velocity of the falling particles.

Calculating the mean value of the vertical velocity in each cell and then labeling each cell as interesting or non-interesting on the basis of this quantity generates a lot of noise. Therefore, we use a procedure which collects a number of cells and examines them collectively. This objective is achieved by randomly generating a large number of rectangular masks labeled by an index m each containing several cells.

Each mask m is a rectangle depending on three parameters, one specifying the coordinates of its bottom-left corner, (i_m, j_m) according to the rule (4.1), the other two the length of its sides in the x- and y-directions, l_{x_m} and l_{y_m}. Each of these parameters is randomly selected. For the bottom-left corner coordinates of each mask, (i_m, j_m), we generate a random integer between 0 and $(N_x \times N_y) - 1$. This integer refers to the label number of a cell which is located in the bottom-left corner of the mask. Then using array C defined in equation 4.3, we can find (i_m, j_m). For instance, if the random integer is 23, then, $i_m = C[23, 1]$ and $j_m = C[23, 2]$ for $N_x = 15$ and $N_y = 10$. The parameter l_{x_m} is a random integer between 1 and L_x, and the parameter l_{y_m} is a random integer between 1 and L_y. Parameters l_{x_m} and l_{y_m} are integer number of cells, which means each mask m consists of an integer number of cells equal to $l_{x_m} \times l_{y_m}$. Finally, using N_m random iterations we create N_m rectangular masks. Figure 4.5 shows four random masks that are generated for the region showed in figure 4.1. For instance, mask m_2 is generated by three random numbers, label number 76, which

refers to c_{76}, $l_{x_{m_2}} = 4$ and $l_{y_{m_2}} = 3$. The bottom-left corner coordinates of the mask m_2, i.e., (i_{m_2}, j_{m_2}), are the bottom-left corner coordinates of the cell c_{76}, which from array C are $(5, 1)$.

The process of mask generation must not be such as to bias specific subregions of R, in other words, every cell should be covered approximately an equal number of times. The vertical coordinate of each data point in figure 4.2 indicates the number of cells that belong to a mask as many times as the corresponding value on the horizontal axis. For the example shown the total number of masks generated was 20000 and the average number of hits is approximately 2764 with a standard deviation of only 44. A robust measure of the dispersion of the data about the mean is the coefficient of variation defined by the ration of the standard deviation to the mean. In this case the coefficient of variation is about 1.6% which is characteristic of a small dispersion. Figure 4.3 is similar but it has been obtained with a small number of masks, i.e., 1000. In this case, the average number of hits is 144 with a standard deviation close to 10 and a coefficient of variation of 7%. On the other hand, figure 4.4 has been obtained with 50000 masks. In this case, the mean number of hits is 6952 with a standard deviation of 62 which is 0.9% of the mean. In view of the marginal improvement over the results obtained with 20000 masks, we have decided that, for the purpose of avoiding biases, 20000 masks are sufficient and this is the number that we have used in our study.

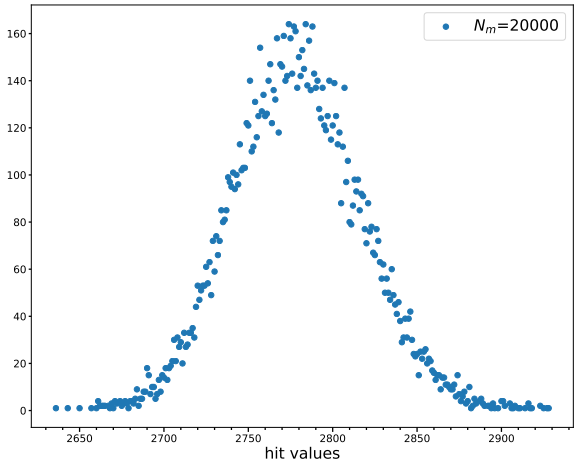


Figure 4.2: Distribution of total number of cells that have been covered a certain number of times by generating 20000 random masks

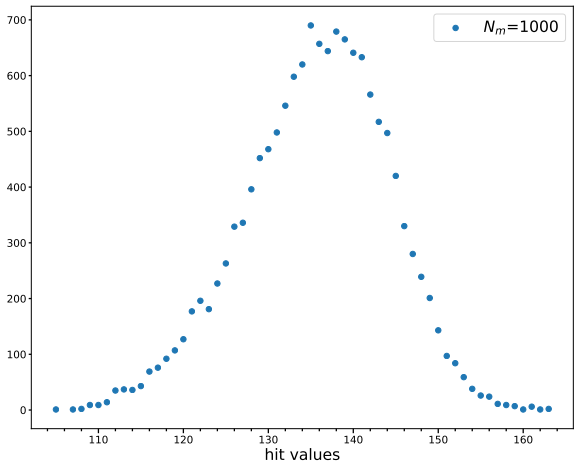


Figure 4.3: Distribution of total number of cells that have been covered a certain number of times by generating 1000 random masks

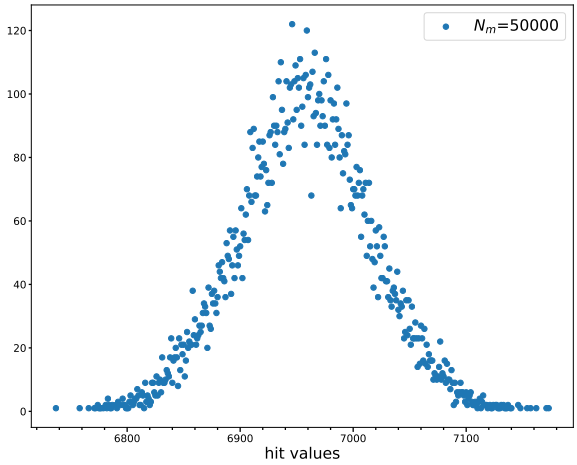


Figure 4.4: Distribution of total number of cells that have been covered a certain number of times by generating 50000 random masks

Since each mask can be as large as the entire domain, it may happen that a mask extends beyond the boundaries of the region R. In this case, we use periodicity to examine the periodic images of the particles located outside the boundaries of region R.

Each mask contains a number of cells. For each rising particle i in each one of these cells, we calculate

$$\frac{|w_{z_p} - \overline{W}_z^+|}{D^+}, \tag{4.4}$$

in which w_{z_p} is the vertical velocity of the particle i. The number of rising particles for which this fraction exceeds a pre-assigned value α is denoted by n_m^+. The choice of α is an important aspect of our method because if it is taken too small, too many particles become potentially interesting whereas, if it is taken too large, there is a possibility of failing to identify particle clusters. In our work, we have used a trial and error method which suggested that the optimal value of α is between 1 and 1.5.

Let the total number of rising particles in the mask m be N_m^+. If the ratio n_m^+/N_m^+ exceeds a pre-assigned value β then the mask m is labeled as interesting and

the value 1 is assigned to a logical predicate $f(m)$ associated with it; other masks have $f(m) = 0$. The same procedure is followed for the falling particles. The parameter β is another user specified quantity of our method. If it is taken too small, very few masks would qualify as interesting. Conversely, if it is taken too large, many masks will qualify as interesting even though they do not contain interesting particles. In our work, the value $\beta = 0.2$ has proven useful.

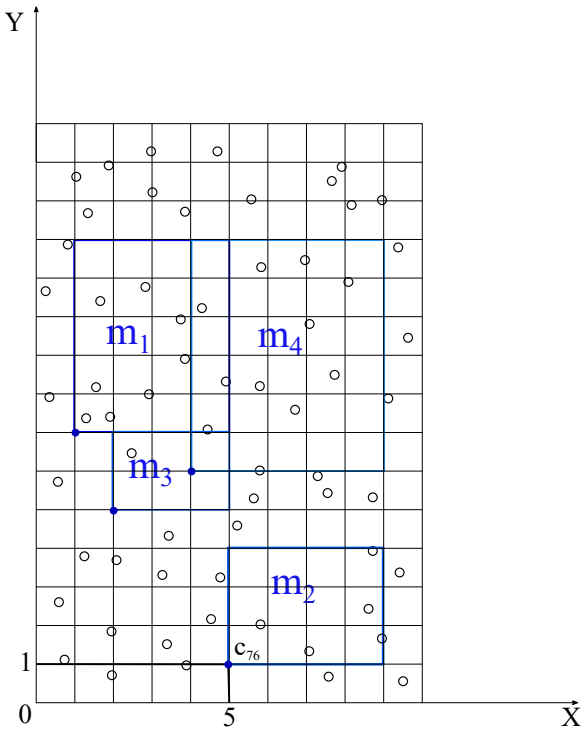


Figure 4.5: Example of some random masks generated for the region shown in figure 4.1

For each cell in the region R we define a counter $h_l^{\pm}$ which starts at 0 and is incremented by 1 every time the cell is part of an interesting mask. In other words, every time cell l is covered by an interesting mask, its counter $h_l^{\pm}$ is incremented by 1. After N_m iterations, the counter of each cell contains information on the number of times that cell was part of an interesting mask. This information is included in the

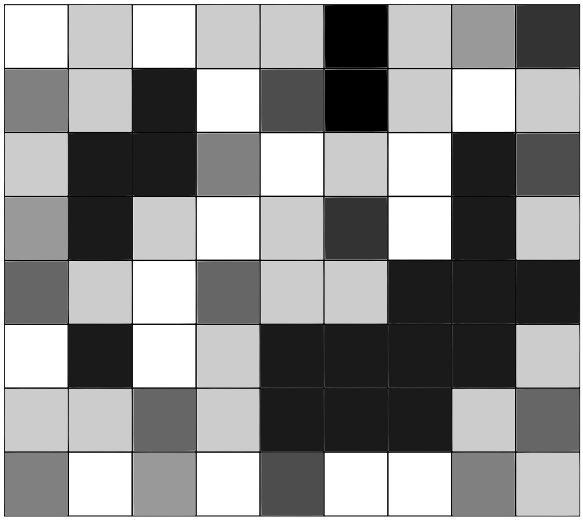


Figure 4.6: Representation of region R in grayscale after assigning hit values to each cell

array C by adding two columns as shown here

$$C = \begin{bmatrix} 0 & 0 & 0 & h_0^+ & h_0^- \\ 1 & 0 & 1 & h_1^+ & h_1^- \\ 2 & 0 & 2 & h_2^+ & h_2^- \\ \vdots & \vdots & \vdots & & \\ (N_x \times N_y) - 2 & L_x & L_y - 1 & h^+_{(N_x \times N_y)-2} & h^-_{(N_x \times N_y)-2} \\ (N_x \times N_y) - 1 & L_x & L_y & h^+_{(N_x \times N_y)-1} & h^-_{(N_x \times N_y)-2} \end{bmatrix}. \tag{4.5}$$

The counters $h_l^{\pm}$ can be considered equivalent to the pixel intensity in the procedure adopted in image processing. The histogram of the counters values is also equivalent to a grayscale level histogram of an image. The process of generating random masks in order to assign a value to the counter for each cell can be thought of as converting the region R into a grayscale image. Figure 4.6 shows an example of the region R with the color assigned to each cell representing its corresponding h_l. The darker the shade of the cell, the larger the counter value is.

4.1.1 Binarizing the region R

The next step is to identify the cells with high values of the counters $h_l^{\pm}$ because these cells potentially constitute clusters. For this purpose, the important attribute is not the actual value of $h_l^{\pm}$ but only whether this value is large enough. In other words, we need to binarize the region R identifying each cell simply as 'black' or 'white', i.e., interesting or non-interesting. This objective is achieved by introducing a parameter T which is used as a threshold to identify interesting cells. For each cell l we define another logical predicate f_c which is assigned the value 1 when $h_l^{\pm}$ exceeds T and 0 in the opposite case

$$f_c(l) = \begin{cases} 1 & h_l \geq T \\ 0 & otherwise \end{cases} . \tag{4.6}$$

This procedure effectively binarizes the cells of the region R into 'white' and 'black' cells (see figure 4.7). We decided to the proper value of T by trail and error.

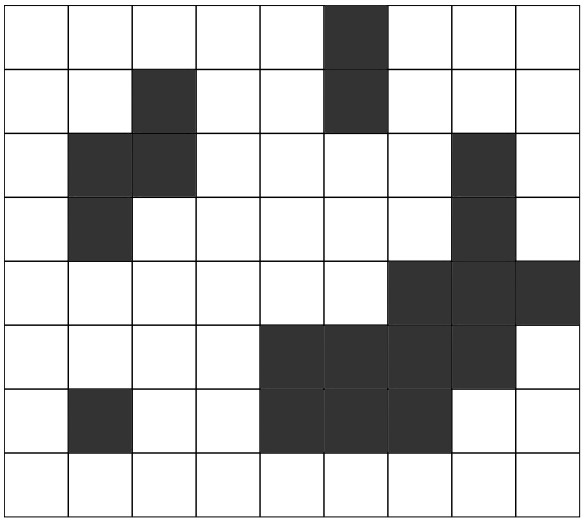


Figure 4.7: Examle shown in figure 4.6 after binarizing by applying threshold T

4.2 Identifying Cell Clusters

While the cells identified by this procedure may contain particle clusters, one cannot draw this conclusion without a further step because if, for example, an isolated cell has a value $f_c = 1$, this could be simply due to random statistical fluctuations. Real clusters must be constituted by a number of adjacent interesting cells. To resolve this point we follow a method inspired by the region growing and region merging methods of image processing. We start from the cell c_0 in the lower left corner of the region R and examine the value of its logical predicate. If this value is 0, we move on to the cell c_1 and do the same thing until we find a cell, with index l, say, for which the predicate is 1. This cell is the potential "seed" for the first cluster $\mathcal{C}_0$ and we assign provisionally the index of this cluster, i.e., 0 in this case, to a new label J associated with each cell. Next we examine the cells to the left and above c_l. In other words, if the first interesting cell c_l has coordinates (i, j), we look at the two cells with coordinates $(i+1, j)$ and $(i, j+1)$ or, using the cell numbering system, cells c_{l+1} and c_{l+N_y}. We next examine the logical predicates of these two cells and, if they have value 1, we assign them to the same set with cell c_l. We then examine the neighbors of the cell located at $(i, j+1)$, if their logical predicate is 1, we combine them in a new set which includes the cell located at $(i, j+1)$. Now we examine whether there is a non-zero intersection between the two sets. If so, the two sets are merged into a single set. We move next to the cell located at $(i, j+2)$, examine its neighbors and, if appropriate, we form a new set including the cell located at $(i, j+2)$. This set is then compared with the one based on the cells located at cell $(i, j+1)$ and if the intersection is non-zero, it is merged with the previous enlarged set. Otherwise, it is a new cluster. Proceeding this way, a point is reached at which no interesting cells are contiguous to the last cell. We continue the systematic examination of the cells until we find one which is labeled interesting. This is the potential seed of a new cluster

and so on. At each step of forming a new cluster, we find the intersection between all clusters. This means that if the new cluster intersects with the previously found clusters, it is merged with them. The reason for this additional step is that, as we move up in y-direction, we find the initial seeds for new clusters, which we need to put in the cluster where they have common neighbors, i.e, intersection. We also should mention that, this way, the clusters that are connected diagonally at a corner will be identified as distinct clusters. Figures 4.8 and 4.9 illustrate the above-mentioned process for the clusters shown in figure 4.7.

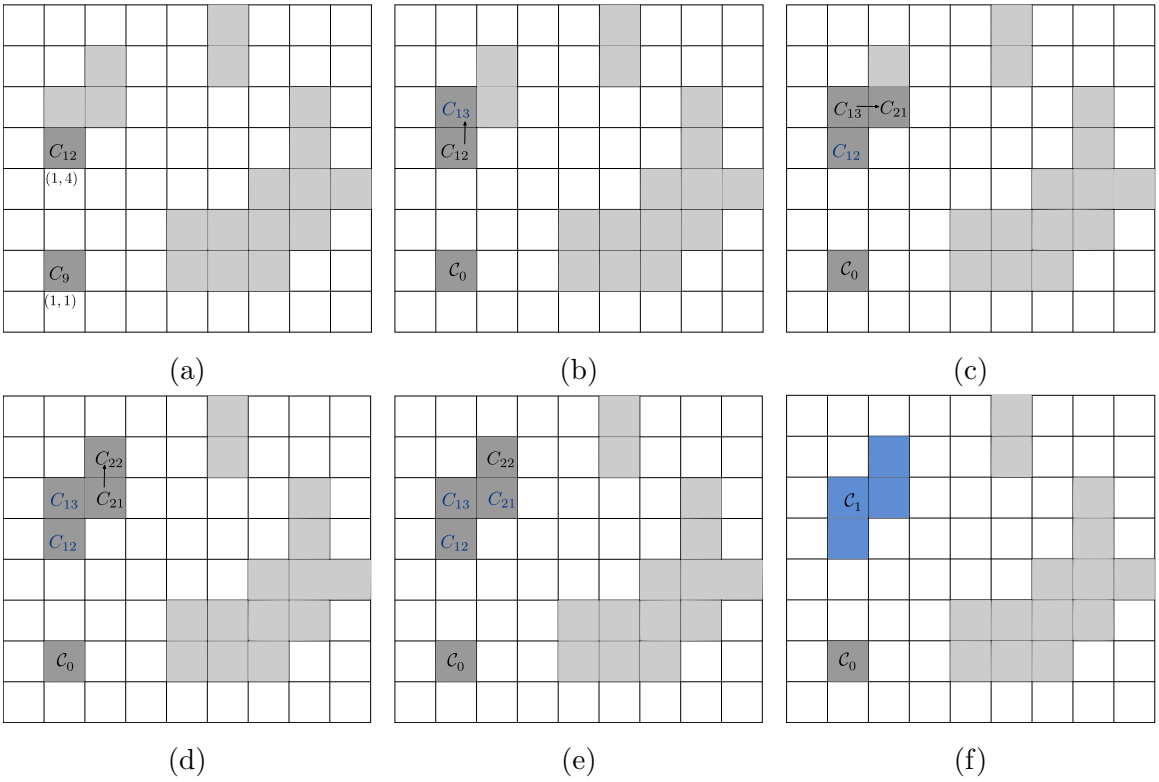


Figure 4.8: Implementation of region growing method for cluster $\mathcal{C}_1$ of figure 4.7

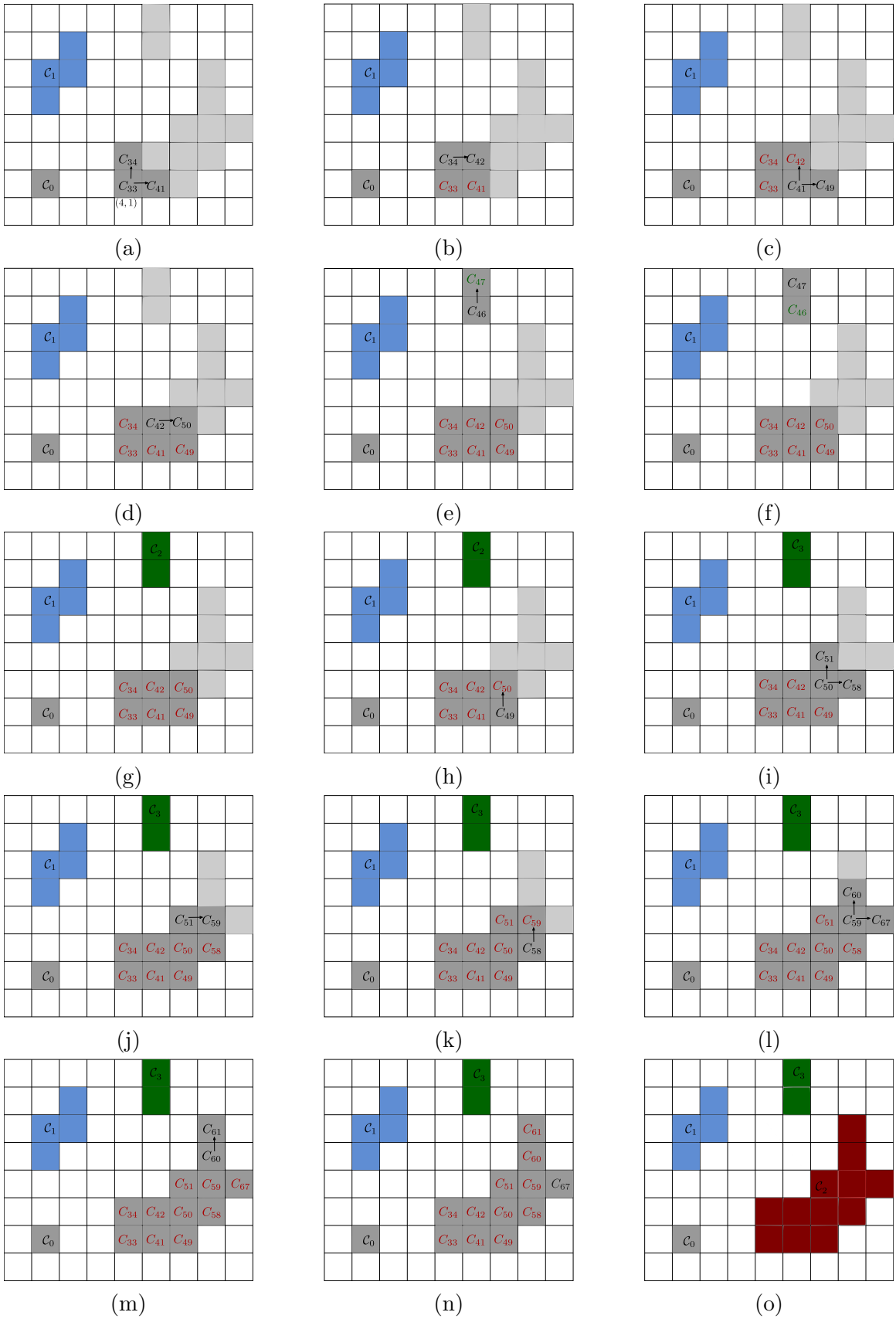


Figure 4.9: Implementation of region growing method for cluster $\mathcal{C}_2$ and $\mathcal{C}_3$

As shown in figure 4.8a, the first interesting cell is c_9 and is located at $(1, 1)$. Neither one of its two neighbors is interesting which makes it effectively an isolated cell, but, provisionally we assign it to a cluster $\mathcal{C}_0$ with a provisional index 0. Then, as we move by 1 increment in the y-direction, the next interesting cell is c_{12} and is located at $(1, 4)$. Since this cell is not connected to cell c_9, we assign it to a new cluster $\mathcal{C}_1$ with the provisional label 1. We then check the logical predicate $f_c(l)$ for the above and the right neighbors of cell c_{12}, i.e., c_{13} and c_{20}. Since $f_c(13) = 1$, we create the set $[c_{12}, c_{13}]$, which consists of the cells belonging to cluster $\mathcal{C}_1$ (figure 4.8b). The next interesting cell is c_{13}, starting from which we create a new set $[c_{13}, c_{21}]$ by checking the logical predicate $f_c(21)$ and $f_c(14)$ (figure 4.8c). Because the intersection of the two sets is non-zero, we grow the cluster $\mathcal{C}_1$ by adding the cell c_{21}. At this point the cluster $\mathcal{C}_1 = [c_{12}, c_{13}, c_{21}]$. Proceeding in this way, the cells c_{12}, c_{13}, c_{21}, c_{22} will be assigned to $\mathcal{C}_1$ (figures 4.8d and 4.8e). The next interesting cell is c_{33}, which potentially belongs to a new cluster $\mathcal{C}_2$ (figure 4.9a). Using the above procedure (figures 4.9a - 4.9d), the cells c_{33}, c_{34}, c_{41}, c_{42}, c_{49} and c_{50} belong to the cluster $\mathcal{C}_2$ and will be provisionally assigned the index 2. The next interesting cell is c_{46} (figures 4.9e), because the neighbors of this cell does not interesect with cells in the previously found clusters, this cell belongs to a new cluster $\mathcal{C}_3$ and will be provisionally assigned the index 3. The next interesting cell is c_{47}, which also belongs to $\mathcal{C}_3$ (figures 4.9f). This way, the interesting cells with the green color in figure 4.9g, i.e., c_{46} and c_{47}, will be assigned to cluster $\mathcal{C}_3$ and will be provisionally indexed as 3. The next interesting cell is c_{49} (figures 4.9h), because this cell and its neighbor, i.e., c_{50}, interesect with the cells in cluster $\mathcal{C}_2$, the cells c_{49} and c_{50} will be assgined to cluster $\mathcal{C}_2$. This is the reason why it is very important to find the interesection betwwen clusters whenever a new cluster is formed. Proceeding with the same way, all the cells shown with with red color in figure 4.9o will be assgined to cluster $\mathcal{C}_2$. We append a column to the array C

which includes the provisional cluster index assigned to each cell. The non-interesting cells are assigned the label NaN. For the interesting cells of the example shown in figure 4.9o the array C will now look as shown in 4.8

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & nan \\ \vdots & \vdots & \vdots & \ddots & nan \\ 9 & 1 & 1 & \cdots & 0 \\ 10 & 1 & 2 & \cdots & nan \\ 11 & 1 & 3 & \cdots & nan \\ 12 & 1 & 4 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 47 & 5 & 7 & \cdots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 60 & 7 & 4 & \cdots & 2 \\ 61 & 7 & 5 & \cdots & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}. \tag{4.7}$$

The next step is the identification and removal of isolated interesting cells and trimming the clusters. Trimming the clusters is defined as eliminating the cells of the clusters that have a number of links to neighboring cells smaller than a pre-assigned value K. In order to eliminate the cells with fewer number of links, for each interesting cell, we look at the 8 neighbors of that cell. If the cell has less than K number of neighbors, we eliminate it from the cluster, i.e., change its current cluster index to NaN in the array C. We continue looking for the interesting cells with K number of links or less until all interesting cells have at least more than K number of links in the 8-adjacent neighborhood. We have chosen K using a trial and error method leading to

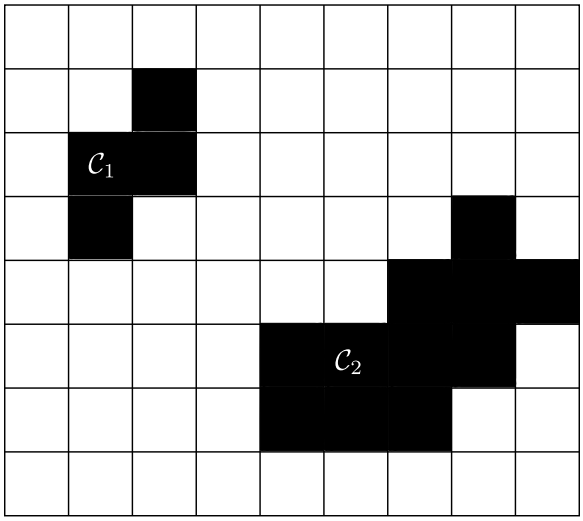


Figure 4.10: Clusters of figure 4.9o after trimming and removing isolated interesting cells

a optimal value between 4 and 8. For clusters shown in figure 4.9o, first, the isolated cell c_9 which belongs to cluster $\mathcal{C}_0$ will be removed. Next, we need to trim the clusters $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$. For the purpose of the illustration, we use $K = 2$ in this example. Thus, the cell c_{61} will be eliminated from cluster $\mathcal{C}_2$. Continuing the trimming process, the cells c_{46} and c_{47} will be removed, which results in removing cluster $\mathcal{C}_3$. Figure 4.9o shows the clusters after removing the isolated cells and trimming. Furthermore,

the array C become

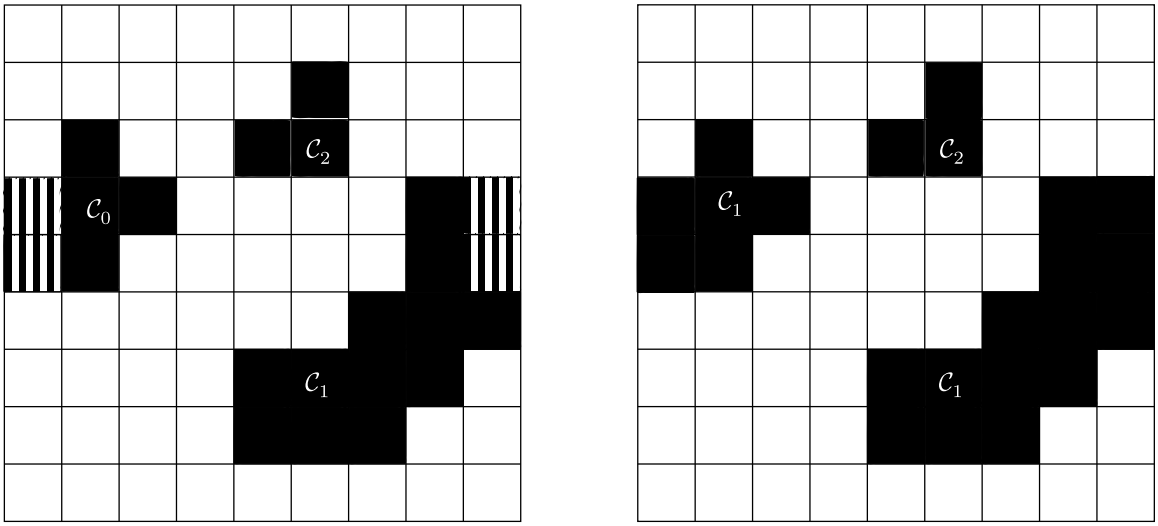
$$\begin{bmatrix} 0 & 0 & 0 & \cdots & nan \\ \vdots & \vdots & \vdots & \ddots & nan \\ 9 & 1 & 1 & \cdots & nan \\ 10 & 1 & 2 & \cdots & nan \\ 11 & 1 & 3 & \cdots & nan \\ 12 & 1 & 4 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 47 & 5 & 7 & \cdots & nan \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 60 & 7 & 4 & \cdots & 2 \\ 61 & 7 & 5 & \cdots & nan \\ \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}. \tag{4.8}$$

4.3 Dealing with Periodicity

As explained in Chapter 2, our data is periodic in the two (or three) dimensions, and it may happen that a cluster straddles the boundary of the region R. For this reason, the previous procedure needs to be suitably modified.

The problem potentially arises only when an interesting cell lies on the boundary of R, i.e., if the cell is located at (i, j) with $i = 0$ *or* $L_x - 1$ or $j = 0$ *or* $L_y - 1$. For example, a cell located at $(i, 0)$, by periodicity, is adjacent to the cell $(i, L_y - 1)$. By the previous procedure, if it is interesting, this latter cell will have been assigned to a potential cluster (recall that isolated cells are also assigned to clusters). After having identified all the potential clusters in the manner described, we check whether the boundary cells of each cluster have periodic neighbors that belong to other potential clusters. If so, we unify the labels of the connected clusters.

For example, cell clusters $\mathcal{C}_0$ and $\mathcal{C}_1$ shown in figure 4.11a are attached by the hatched cells. Therefore, if such clusters appear for our data, we need to join them. Specifically, by joining the cell clusters, we adjust the provisional label of the connected clusters, in this case, $\mathcal{C}_0$ become $\mathcal{C}_1$ (figure 4.11b). Figure 4.12 shows the cluster $\mathcal{C}_1$ has been translated to the opposite boundary because of the periodicity.



(a) (b)

Figure 4.11: An example of cell clusters that are attached on the opposite boundary of region R

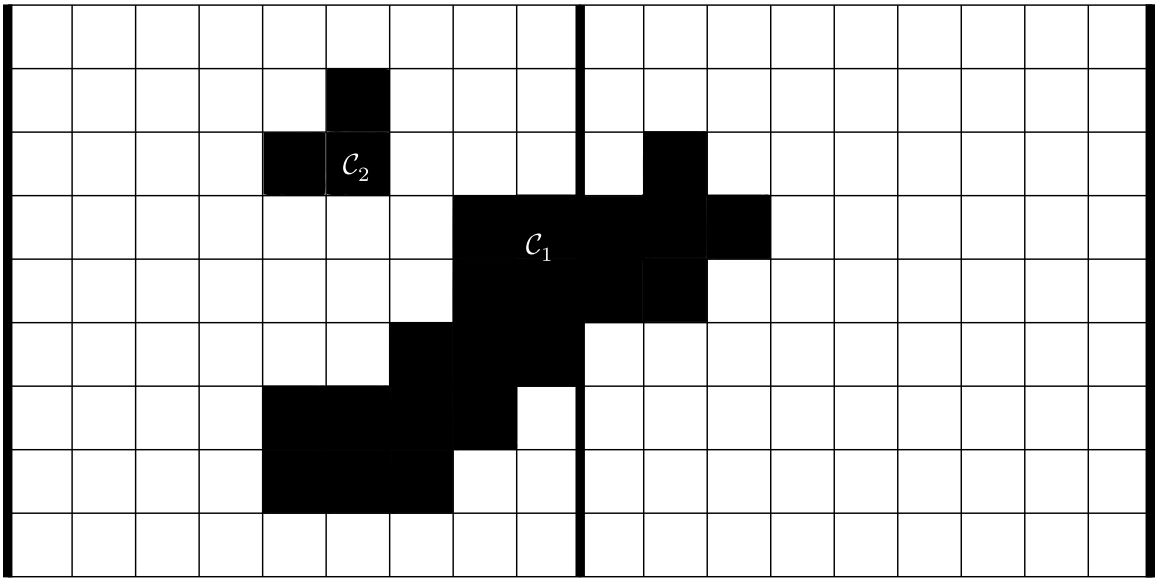


Figure 4.12: Cell clusters of figure 4.11 after being joined and translated to the opposite boundary

The procedure for joining the boundary clusters is as follows: for each cluster, we start from the first cell of the cluster, i.e., the cell in the bottom-left corner, and then we loop over all the cells. If there is a boundary cell which is labeled as c_l, and if there is any cell in any other cluster with the index $c_{l\pm N_y-1}$ or $c_{l\pm(N_y-1)\times(N_x-1)}$, we connect these clusters together. Then, we repeat the procedure for the next cluster.

4.4 Tracking Cell Clusters

Since the objective of present study is to be able to examine the persistence of clusters over time, it is necessary to track them over time. This requires identifying at each time step, the new position of previously identified clusters, or the disappearance of previous clusters or the appearance of new ones. This process is rendered more complex by the periodicity of the system as we now explain.

Following the description presented in the previous section, we have assigned

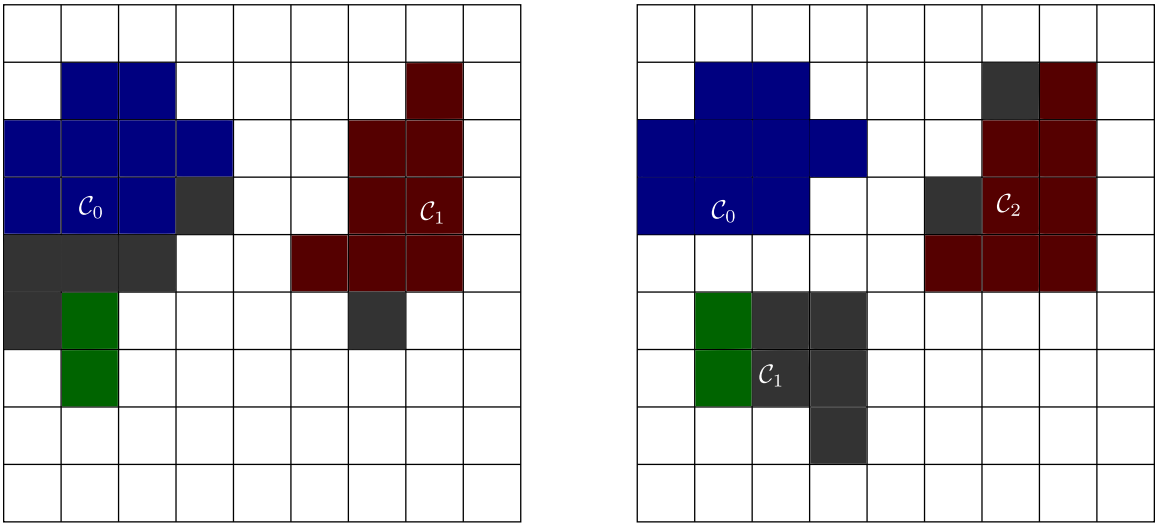
provisional labels to each cluster. For the initial time step, i.e., t_0, these labels are actually final. We, then, consider the next time step, i.e., t_1, and assign the provisional labels by the same method. In order to finalize these labels, we need to examine whether one or more clusters identified at t_1 are translated or modified versions, i.e., evolutions, of the clusters of the previous time step. Figure 4.13 shows an example where clusters at t_1 and t_0, though clearly related, carry different labels. The process of tracking clusters requires that labels of a cluster at t_1 that is the evolution of a cluster at t_0 be made the same as those of the earlier cluster. The same procedure is followed to connect the clusters at t_2 to those at t_1 and so on.

The method by which we carry out this step is based on the premise that a cluster at t_1 which has evolved from a cluster at t_0, most likely will be constituted by many cells which are also part of the earlier cluster. To quantify this notion, for each pairs of clusters at t_0 and t_1, we define a membership ratio as the number of cells that the two clusters have in common divided by the total number of cells of the earlier cluster. If the number of clusters at t_1 is equal to or smaller than the number of clusters at t_0, each cluster at t_1 is associated to the cluster at t_0 which has the largest value of the membership ratio. The provisional labels assigned to its cells are set equal to those of the earlier cluster. These are now the final values of the labels at time step t_1.

If the number of clusters at t_1 is larger than that at t_0, this means that one or more new clusters have formed at time step t_1. In this case, the previous procedure needs to be slightly modified to identify the right pairings. The method of identifying new clusters is as follows. We first find the maximum membership ratio, r_{M_J}, of each cluster J at t_1 with all the clusters at t_0. Then, we find the smallest value of all these maximum membership ratios, r_m. The new cluster is the one the index J is such that $r_{M_J} = r_m$. The remaining clusters are the ones that evolved from the previous time

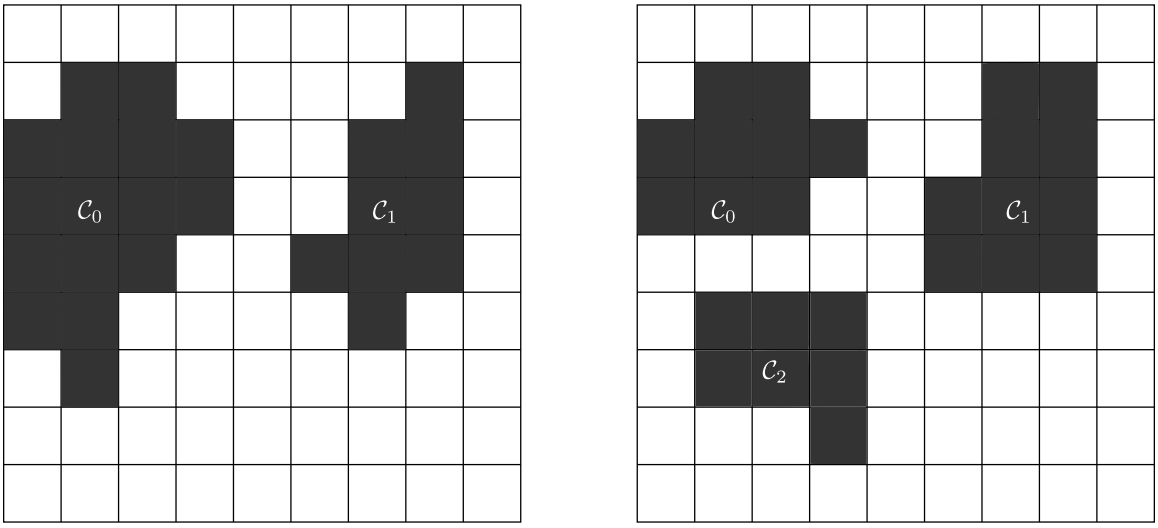
step t_0, and can be paired using the previous procedure.

Figure 4.13 illustrates the above procedure. As we can see from this figure, a new cluster, i.e., $\mathcal{C}_1$ has appeared at time step t_1, and the clusters $\mathcal{C}_0$ and $\mathcal{C}_2$ at time step t_1 have been evolved from clusters $\mathcal{C}_0$ and $\mathcal{C}_1$ at time step t_0, respectively. Following the procedure, since the number of clusters at t_1 is larger than at t_0, it means a new cluster has formed. The new cluster, in this example is $\mathcal{C}_1$, and its cells will be assigned a final label of 2. The remaining clusters at t_1, i.e., $\mathcal{C}_0$ and $\mathcal{C}_2$, will be renumbered as $\mathcal{C}_0$ and $\mathcal{C}_1$ and their cells will be assigned the corresponding labels of the previous time step.



(a) t_0 (b) t_1

Figure 4.13: An example of cell clusters that are assigned different provisional labels in time steps t_0 and t_1



(a) t_0 (b) t_1

Figure 4.14: The clusters of figure 4.13 at time step t_0 and t_1 after assigning the final labels for clusters at t_1

4.5 Identifying Particle Clusters

Now that we have found cell clusters, the final step is to find the clusters of the rising and the falling particles. To proceed with this step, we take the integer part of the x- , y- and z-coordinates of each particle p, which give us the coordinates of the bottom-left corner of the cell to which the particle p belongs. Using equation 4.2 we can find the label number of the cell corresponding to this particle. For example, from figure 4.15, particle p_r is located at $(1.7, 1.2)$, the integer part of the particle p_r coordinate is equal to $(1, 1)$ which corresponds to the cell with the bottom-left corner coordinates of $(1, 1)$. Using equation 4.1, the label number for this cell is c_9. Then, from array C, we will check if this cell is an interesting cell, which in this case it is not. Looking at all particles, when a particle is inside an interesting cell, we then assign a label to that particle which is the same label as that of the cell. At the end of this process, all groups of particles which have the same label form the particle clusters. For example, from the figure 4.15, the blue particles will be assigned an index 0 and

the red particles will be assigned an index 1 as for the label of the corresponding cell cluster. We can also see from this figure that clusters on the boundary have been unified and copied on the opposite boundary due to the periodicity.

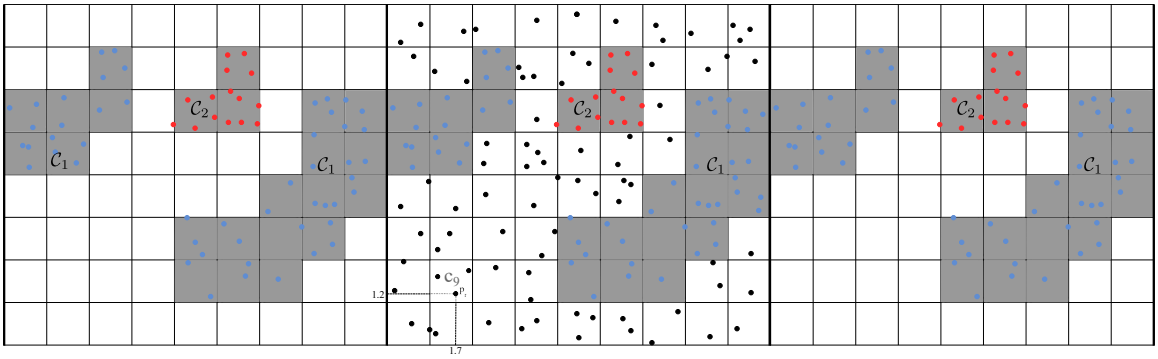


Figure 4.15: Particle clusters in a region R

4.6 Extending to Three Dimensions

In previous sections, for clarity of exposition, we have described our method in the case of two dimensions assuming R is a two-dimensional region. The extension to three-dimensional requires some modification, therefore, in this section, we explain how the various steps are modified to three-dimensional space.

The first step, is identifying the interesting cells. Like the two-dimensional case, we start by dividing the region R into cells. Therefore, we divide the three-dimensional region R with volume $L_x \times L_y \times L_z$ into $N_x \times N_y \times N_z$ number of cells. The cell sides $\ell_x = \ell_y = 1/4\ell_z$ are chosen in a such way that, on average, one particle per cell is expected. In other words, the cell size is chosen in such a way that $\ell_x \times \ell_y \times \ell_z$ multiplied by the total number of particles equals the volume of the region R. We use the equation 4.2 for numbering the cells in which i,j and k are the bottom-left coordinates of the cell c_l. Therefore, for the case of three-dimension, the

array C is as follows

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 2 \\ \vdots & \vdots & \vdots & \vdots \\ (N_x \times N_y \times N_z) - 2 & L_x & L_y & L_z - 1 \\ (N_x \times N_y \times N_z) - 1 & L_x & L_y & L_z \end{bmatrix}. \tag{4.9}$$

The next step is generating N_m number of random masks for which we need to generate four random numbers from a uniform distribution. The first number which needs to be a number between 0 and $(N_x \times N_y \times N_z) - 1$ gives us the label number of the cell located at the bottom-left corner of the mask. The next three random numbers which are generated are the lengths of the mask in x-, y- and z- directions, i.e., l_{x_m}, l_{y_m} and l_{z_m}. Then we calculate 4.4 for each rising and falling particle in each of these three-dimensional random mask m. Using pre-assigned values of α and β, we can calculate the logical predicate $f(m)$ for each mask. Therefore, if $f(m)$ is 1, the mask is interesting. Then, we use the same method we described for the case of two dimensions to find the counter value $h_l^{\pm}$ of each cell c_l. The array C then becomes

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & h_0^+ & h_0^- \\ 1 & 0 & 0 & 1 & h_1^+ & h_1^- \\ 2 & 0 & 0 & 2 & h_2^+ & h_2^- \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (N_x \times N_y \times N_z) - 2 & L_x & L_y & L_z - 1 & h^+_{(N_x \times N_y \times N_z)-2} & h^-_{(N_x \times N_y \times N_z)-2} \\ (N_x \times N_y \times N_z) - 1 & L_x & L_y & L_z & h^+_{(N_x \times N_y \times N_z)-1} & h^-_{(N_x \times N_y \times N_z)-1} \end{bmatrix}. \tag{4.10}$$

Thus, using threshold T, we binarize the three-dimensional region R into interesting and non-interesting cells.

The next step is identifying the cell clusters. We utilize the region-growing and merging method we have described in section 4.2 for the case of two dimensions with some modifications.

As for the case of two dimensions, we start from the first cell, c_0 of the region R, with the coordinates $i = 0$, $j = 0$ and $k = 0$. The main modification to three-dimensional space is that when a cell, for example c_l, is interesting, we examine its front, back and left neighbors as well as its above and right neighbors to check if they are interesting or not. For instance, if the interesting cell c_l has the coordinates of (i, j, k), we examine the five neighbors with the coordinates $(i, j, k+1)$, $(i+1, j, k)$, $(i-1, j, k)$, $(i, j+1, k)$ and $(i, j-1, k)$, or using cell numbering systems, the neighbors with the labels c_{l+1}, $c_{l+Nz\times Ny}$, $c_{l-Nz\times Ny}$, c_{l+Nz} and c_{l-Nz} corresponding to the above, front, behind, right and left neighbors respectively. Repeating the same steps of the two-dimensional case, we identify the clusters by giving their cell a provisional label.

In order to handle the periodicity in three-dimensional case, we test if any cluster has at least one boundary cell, i.e., a cell which is located at (i, j) with $i = 0$ *or* $L_x - 1$ or $j = 0$ *or* $L_y - 1$ or $k = 0$ *or* $L_z - 1$. If another cluster has at least one cell on the opposite boundary, i.e, $i = L_x - 1$ *or* $i = 0$ or $j = L_y - 1$ *or* 0 or $k = L_z - 1$ *or* $k = 0$, these two clusters are indeed one cluster and they need to be joined by unifying their labels. The piece of the unified cluster with smaller number of cells is also copied on the other side of the larger piece for visualization purposes.

The procedure for tracking the clusters over time for the case of three dimensions is exactly the same as for the two-dimensional case, because when tracking the clusters, we only work with the cell labels to find the membership ratio between two clusters in two consecutive times steps.

4.7 Connection with Digital Image Segmentation

In this section, we discuss the connection between our method in two-dimensional space and image segmentation and how our method is analogos to the methods we have introduced in Chapter 1. The most significant difference is that in the case of digital image processing, the location and the intensity values of each pixel are known, whereas in our case, we don't have that information *a priori*. Therefore, we need to take an additional step to assign the equivalent of graylevel to the cells into which we have divided the region R. We perform this step by generating N_m number of random masks with uniformly random generated corners and dimensions, these masks are equivalent to the pixel local window of local thresholding. Like the local thresholding methods, by looking at the standard deviation and the mean value of vertical velocity of particles, we decide if the mask is interesting, if so, we give each cell in the mask an increment. The significant difference between the masks in our method and the pixel local windows is that since the masks have different size and corner coordinate, they overlap in a uniform way, therefore a cell will be covered more than once if it belongs to the interesting subregion of the region R. The pixel local windows on the other hand, do not overlap and they have a fixed size. More importantly, our method first assign an interesting label to the masks based on the logical predicate, whereas, local thresholding decide if each pixel is interesting or so-called foreground based on the logical predicate for each stop of window. Therefore, our method avoids generating noise when assigning a value to each cell. After a value equivalent to intensity is assigned to each cell, we use a global threshold over the entire region to binarize it, which means labeling the high intensity cells as interesting and the rest as non-interesting. Thus, we use an integration of local and global thresholding to binarize the region.

After binarizing the region and identifying the interesting cells, we need

to establish the links between these interesting cells. Thus, we use region growing method integrated with region merging to find the clusters in the domain. The clusters are the regions that grow from a cell. For our method, we start systematically from one corner to find the first interesting cell. This is our initial seed in order to start the growth. However, the initial seed in the region growing image segmentation is random and is corrected through the growth process. Since we start systematically from one corner and we check the above and right neighbors, the growth stops when the algorithm reaches the last cell, i.e., the top-right corner. As for the final step, we need to merge the clusters into one if they have intersections.

In the next Chapter, we describe the results of the our methodology. Table 4.5 summarized the parameters value, N_m, α, β, K that we use for our study.

Table 4.5: Pre-assigned parameter values used in our study

parameters	meaning	value
N_m	number of random iterations for generating the random masks	20000
α	particle velocities distance from mean over the standard dviation	1.5
β	ratio of interesting particles in a box	0.2
T	Threshold for binarizing the region R	0.02
K	number of links in the 8-adjacent neighboors	4

Chapter 5

Results

In this Chapter, we first present several examples of the cell and the particle clusters that we have found using our methodology described in Chapter 4. Then, in the next section, we verify our method by comparing the average vertical velocity of the falling and rising particle clusters and that of the entire region R. The results are found using the values described in table 4.5 for parameters, N_m, α, β, T and k.

5.1 Cell and Particle Clusters

Since our data is three-dimensional, we have implemented the methodology in three-dimensional space to derive the following results. However, for better understanding, we first present results in a two-dimensional form obtained by projecting the three-dimensional data onto the xz-plane. We present the same results in three dimensions. In this section, we demonstrate how our method works in the various situations that arise in our data. As we described in Chapter 2, for our current study, the last 501 time steps of the simulation have been selected. Therefore, for an easier presentation of the results, we number the time steps sequentially 0, 1, 2, ..., etc. all the way to 500.

(a) t=335 (b) t=340 (c) t=345

(d) t=350 (e) t=355 (f) t=360

(g) t=365 (h) t=370 (i) t=375

Figure 5.1: Falling cell clusters in xz-plane for time steps $t = 335$ to $t = 375$

(a) t=335 (b) t=340 (c) t=345

(d) t=350 (e) t=355 (f) t=360

(g) t=365 (h) t=370 (i) t=375

Figure 5.2: Falling particle clusters in xz-plane for time steps $t = 335$ to $t = 375$

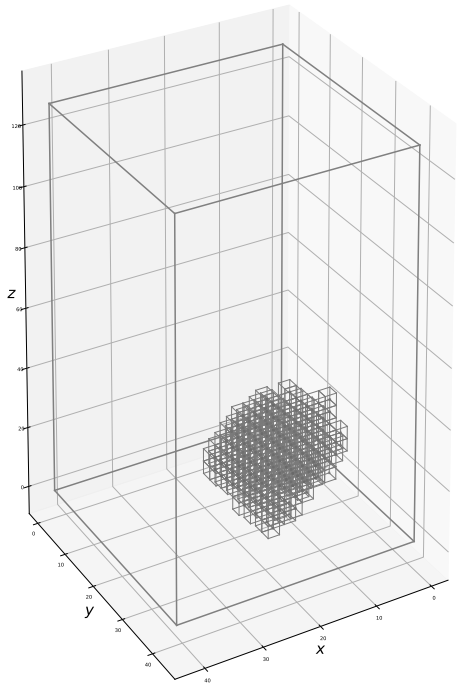


Figure 5.3: The falling cell cluster of time step $t = 335$ in three dimensions

We begin with the simplest situation in which only one cluster exist. As we explained in Chapter 2, the time steps are small in order for the time integration to be accurate. Therefore, the particles displacement over a single time step are small. For this reason, we show the cell clusters every 5 time steps in order to better appreciate their evolution in time. Another thing to keep in mind in evaluating the results is that the mean velocity of all the particles is zero so that the clusters undergo limited displacement either upward or downward. Thus, for example, no cluster will start at the top of the domain and fall all the way to the bottom because it will disintegrate long before the time necessary for such a large displacement.

Figures 5.1a to 5.1i show a projection of a falling cell cluster, i.e., $\mathcal{C}_0$, in xz-plane from time step $t = 335$ to time step $t = 375$. The gray dots represent the cell corners. The cluster shape changes with time as the particles constituting it move and change their velocity. Using the method described in section 4.6, figures 5.2a to 5.2i show the projection of corresponding particle clusters of figures 5.1 in xz-plane.

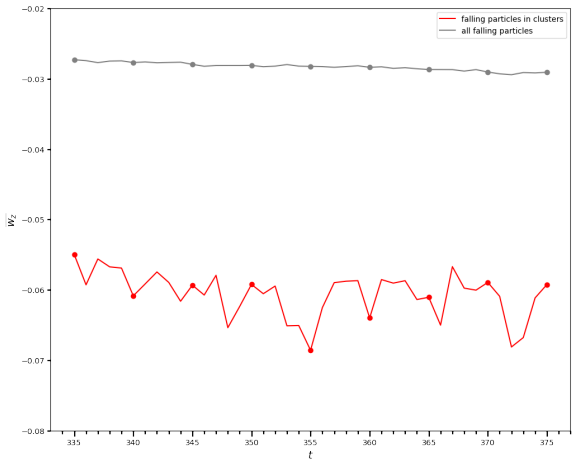
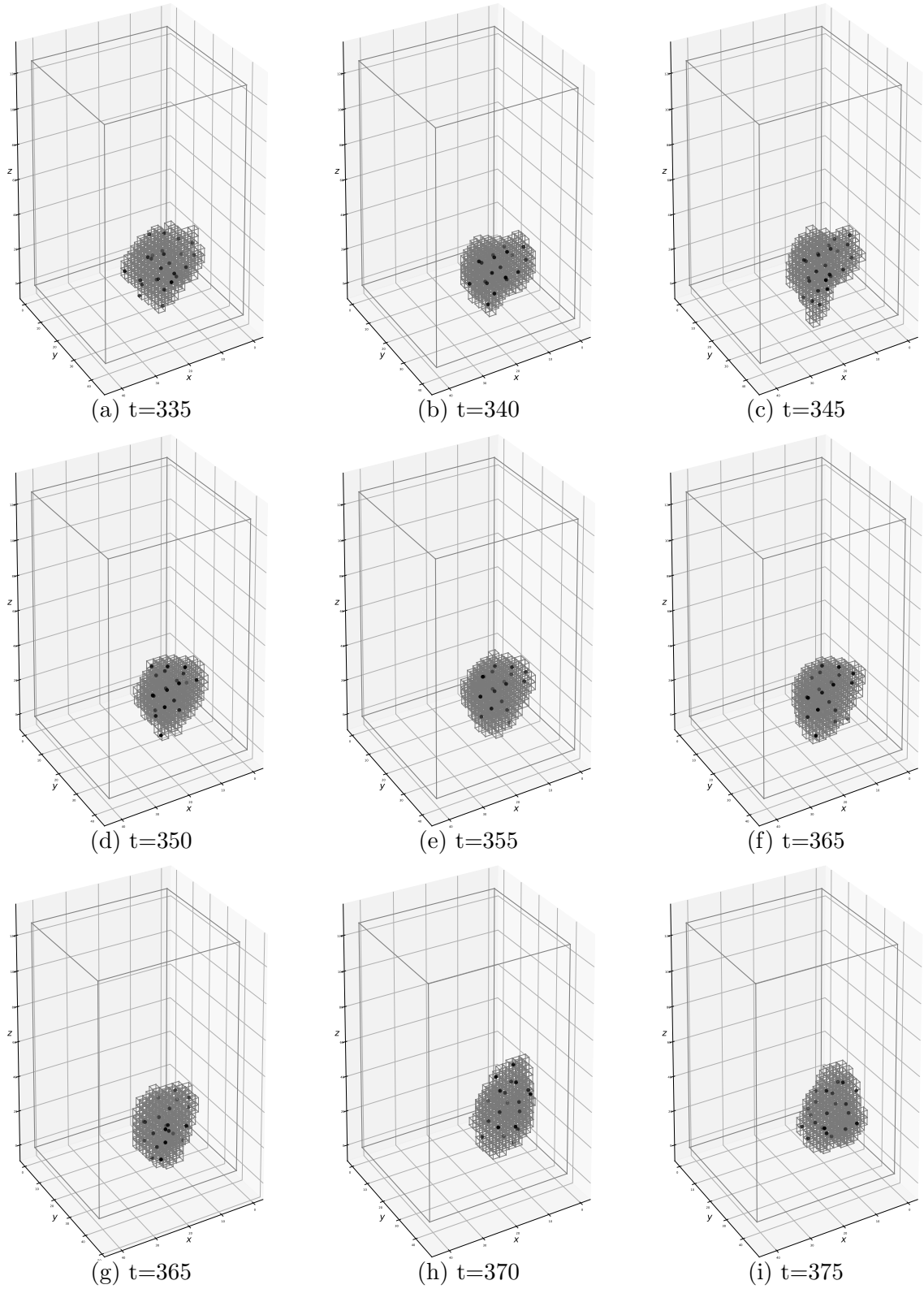


Figure 5.4: Average vertical velocity of falling particles in cluster and all falling particles for time steps $t = 335$ to $t = 375$

The gray circles in these figures are the particles inside the cluster shown in black color.

An example of the three-dimensional rendering of the cell cluster presented in two dimensions in the previous figure for $t = 335$ is shown in figure 5.3. The gray dots in all three-dimensional figures, including this one, represent the corner points of the cells, and the cells with the gray edge color represent the cell cluster with the label of 0, i.e., cluster $\mathcal{C}_0$. A complete three-dimensional sequence of the particle clusters of figure 5.2 is shown in figure 5.5. It can be seen here that the number of particles in the cluster is not constant but fluctuates in time. The average velocity of the particles constituting this particular cluster is shown vs time in figure 5.4 where it is compared the mean falling velocity of all the particles in the region R.

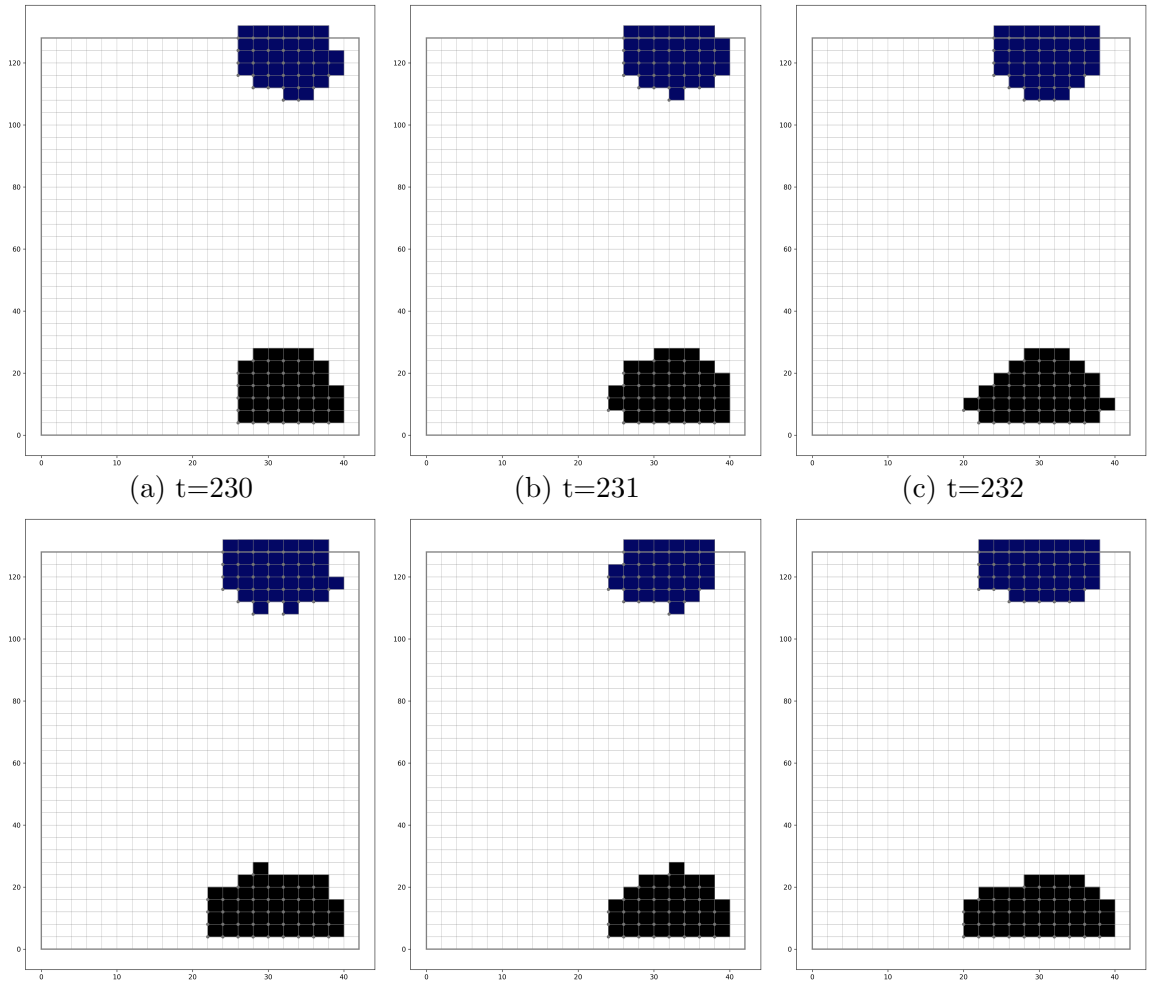


(a) t=335 (b) t=340 (c) t=345

(d) t=350 (e) t=355 (f) t=365

(g) t=365 (h) t=370 (i) t=375

Figure 5.5: Evolution of a falling particle cluster in three dimensions

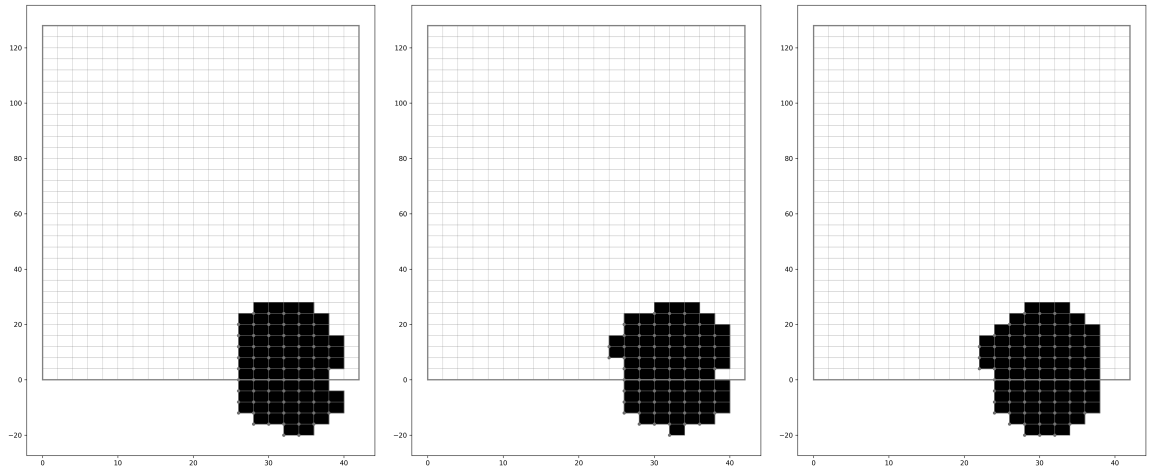


(a) t=230 (b) t=231 (c) t=232

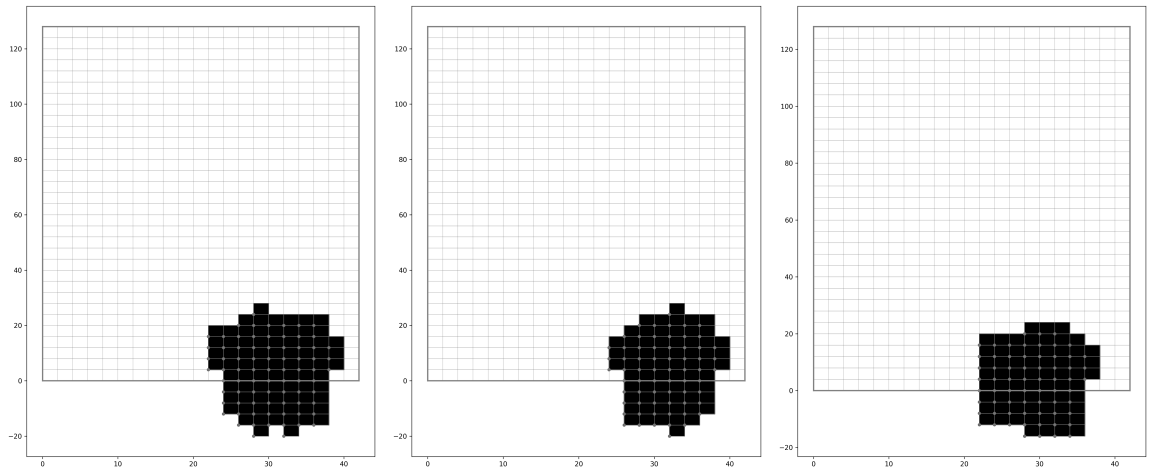
(d) t=233 (e) t=234 (f) t=235

Figure 5.6: Rising cell clusters in xz-plane for time steps $t = 230$ to $t = 235$ before being joined

Figure 5.6 shows a case of rising cell clusters where two clusters have appeared on opposite boundaries because of the periodicity from time steps $t = 230$ to $t = 235$ in xz-plane.



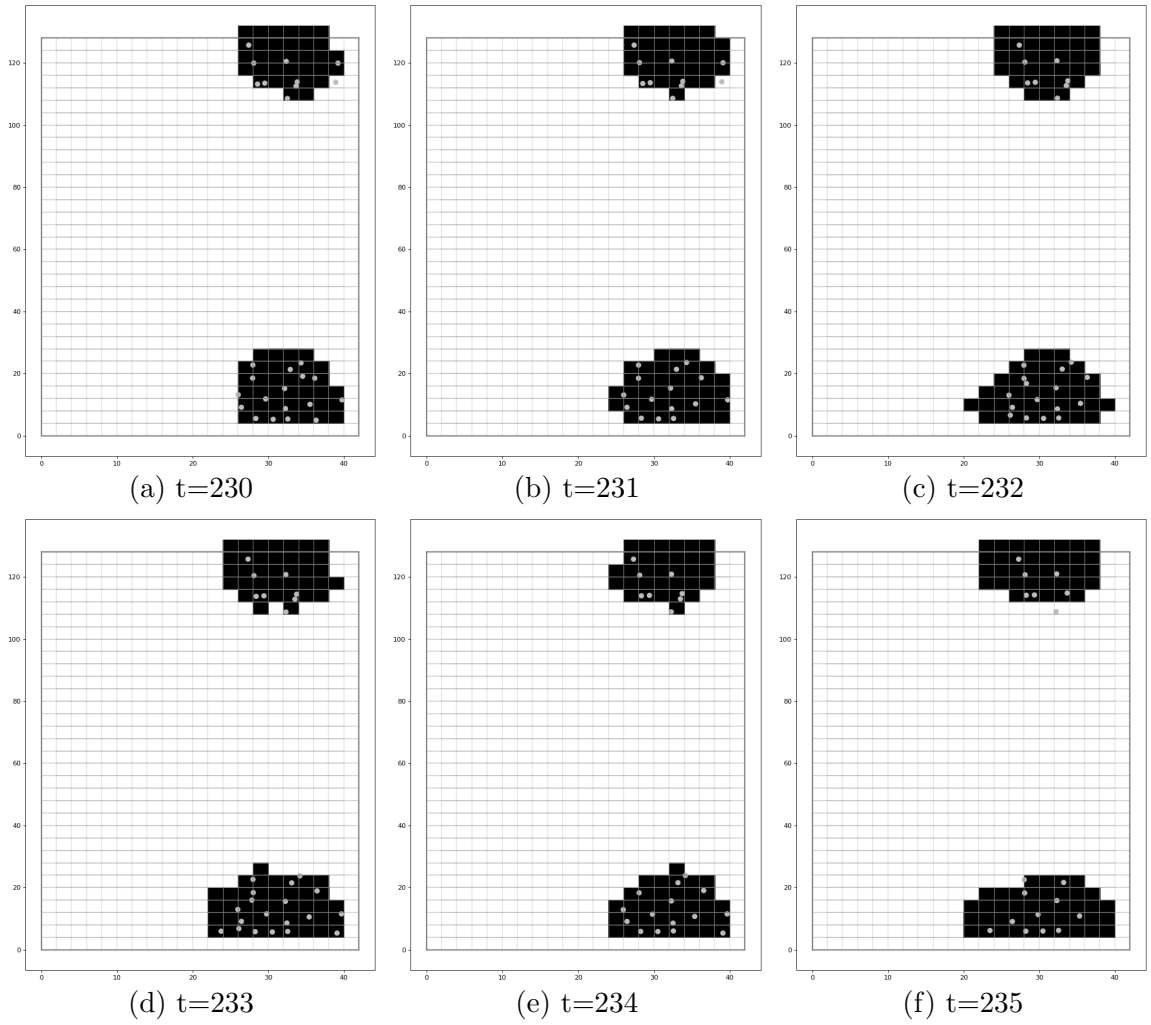
(a) t=230 (b) t=231 (c) t=232



(d) t=233 (e) t=234 (f) t=235

Figure 5.7: Rising cell clusters in xz-plane for time steps $t = 230$ to $t = 235$ after being joined

Using our methodology described in the section 4.6, we join these two clusters into a single one shown in black color in figure 5.7. The evolution of these clusters over six time steps in xz-plane is presented in figures 5.7a to 5.7f.

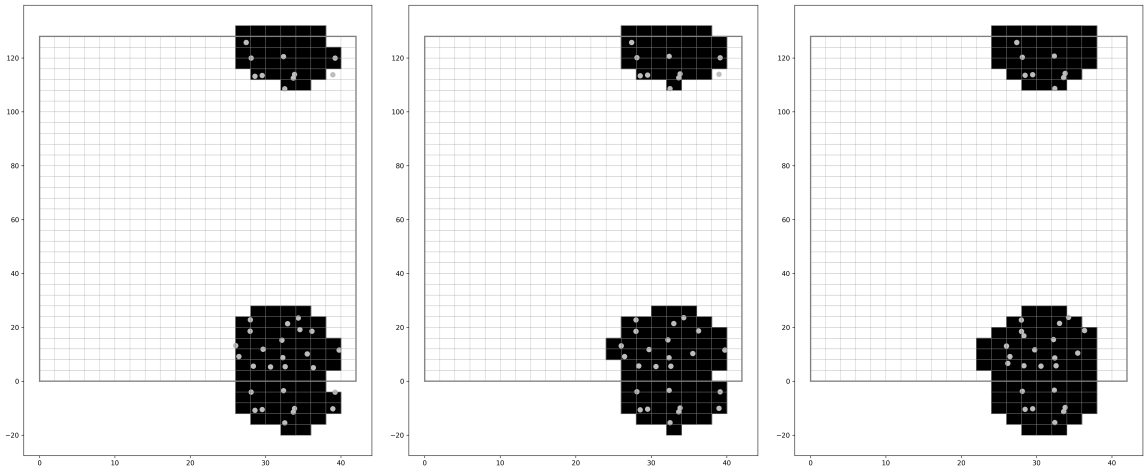


(a) t=230 (b) t=231 (c) t=232

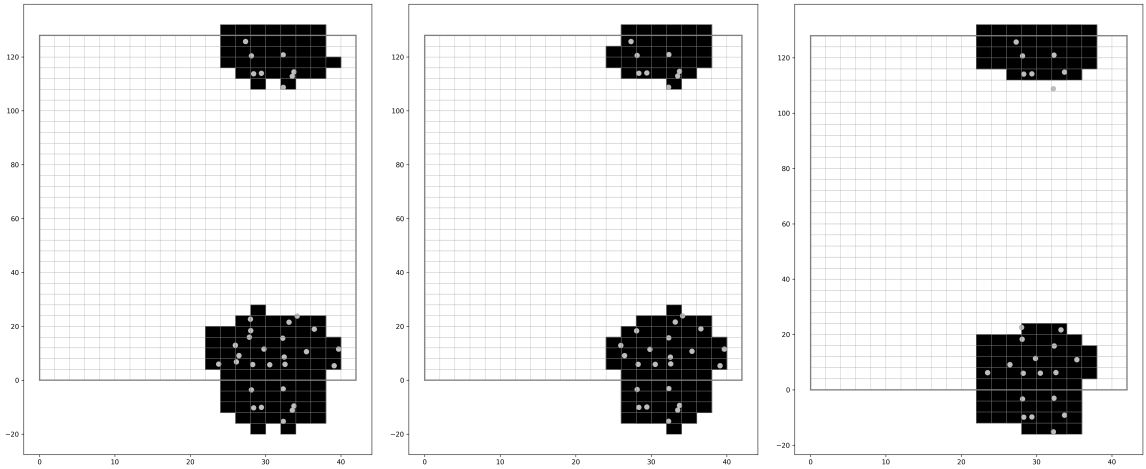
(d) t=233 (e) t=234 (f) t=235

Figure 5.8: Rising particle clusters in xz-plane for time steps $t = 230$ to $t = 235$ before copping on periodic boundary

The corresponding particle clusters before and after joining the two pieces of the cluster are shown in two dimensions in figures 5.8 and 5.9.



(a) t=230 (b) t=231 (c) t=232



(d) t=233 (e) t=234 (f) t=235

Figure 5.9: Rising particle clusters in xz-plane for time steps $t = 230$ to $t = 235$ after joining and being copied on the periodic boundary

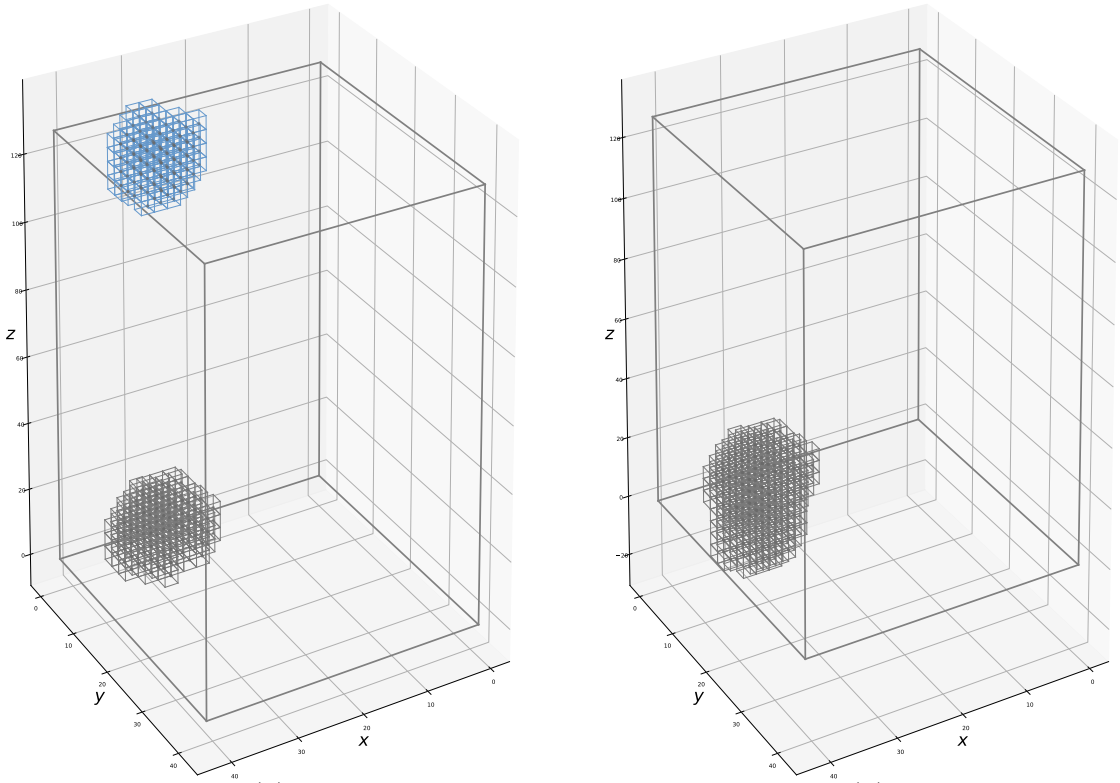
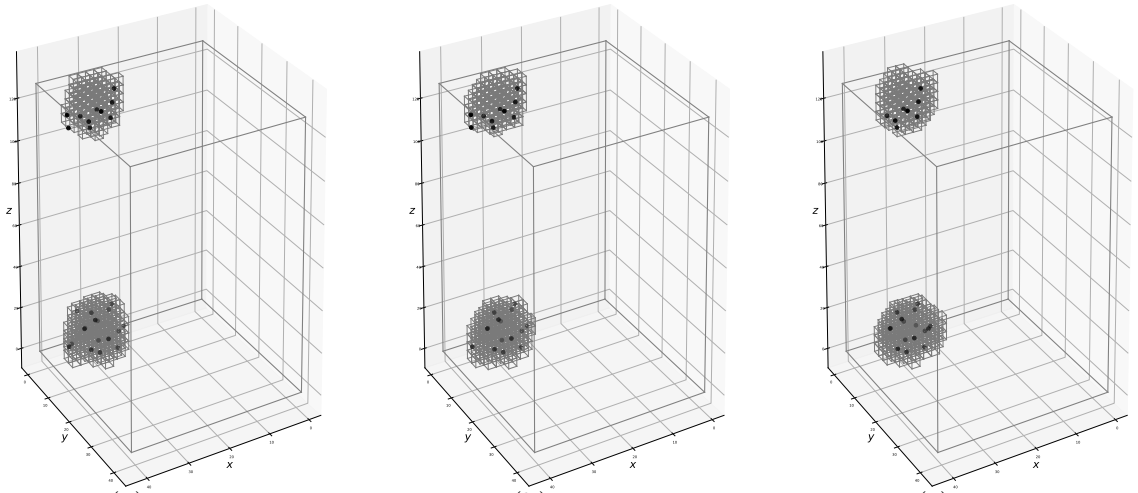
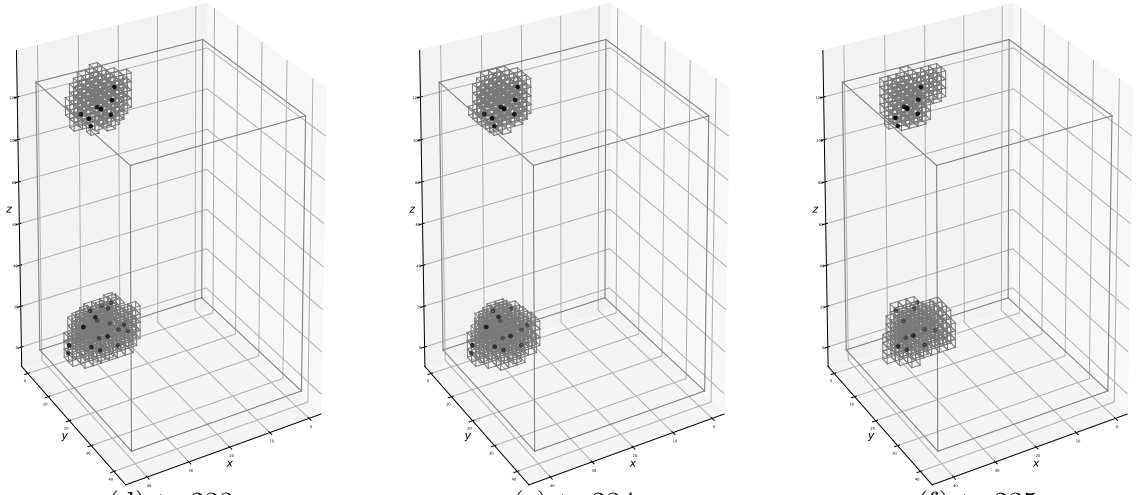


Figure 5.10: Rising cell clusters in three dimensions for time steps $t = 232$ before (a) and after (b) joining

Figure 5.10 shows the cell cluster of time step $t = 232$ before and after being joined in three-dimensional representation. The figure on the left shows the clusters identified by our method before the analysis of boundary cells. After this analysis is completed, the method identifies the two clusters as being in fact two pieces of a single cluster which straddles the boundary of the periodic domain. The two pieces should then be joined with the result shown in the right figure. Figure 5.11 shows a complete sequence of the particle clusters of figure 5.8 in three dimensions before the clusters which are on the boundary have been joined and figure 5.12 shows the same clusters after they have been joined and copied across the periodicity boundary.

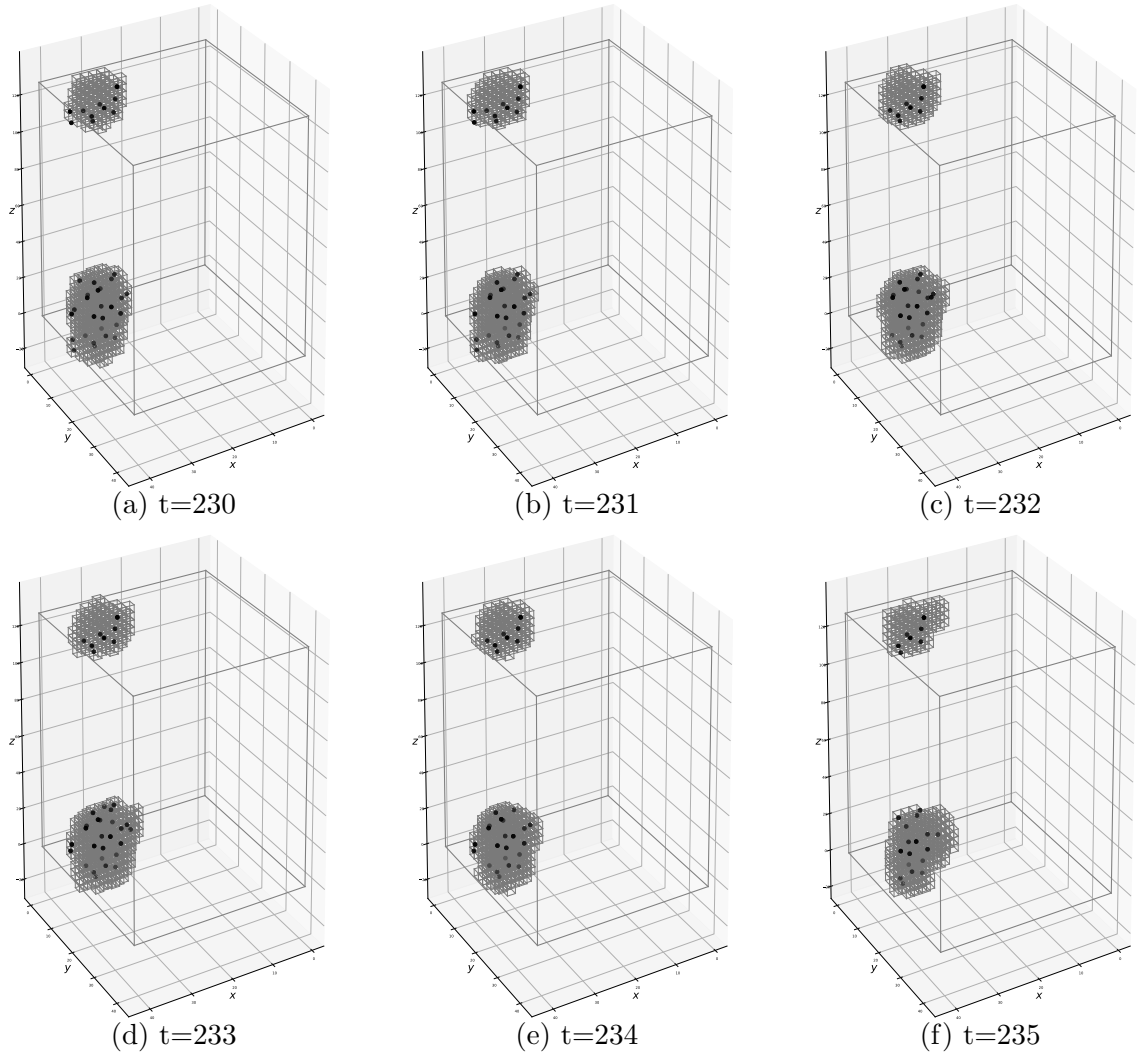


(a) t=230 (b) t=231 (c) t=232



(d) t=233 (e) t=234 (f) t=235

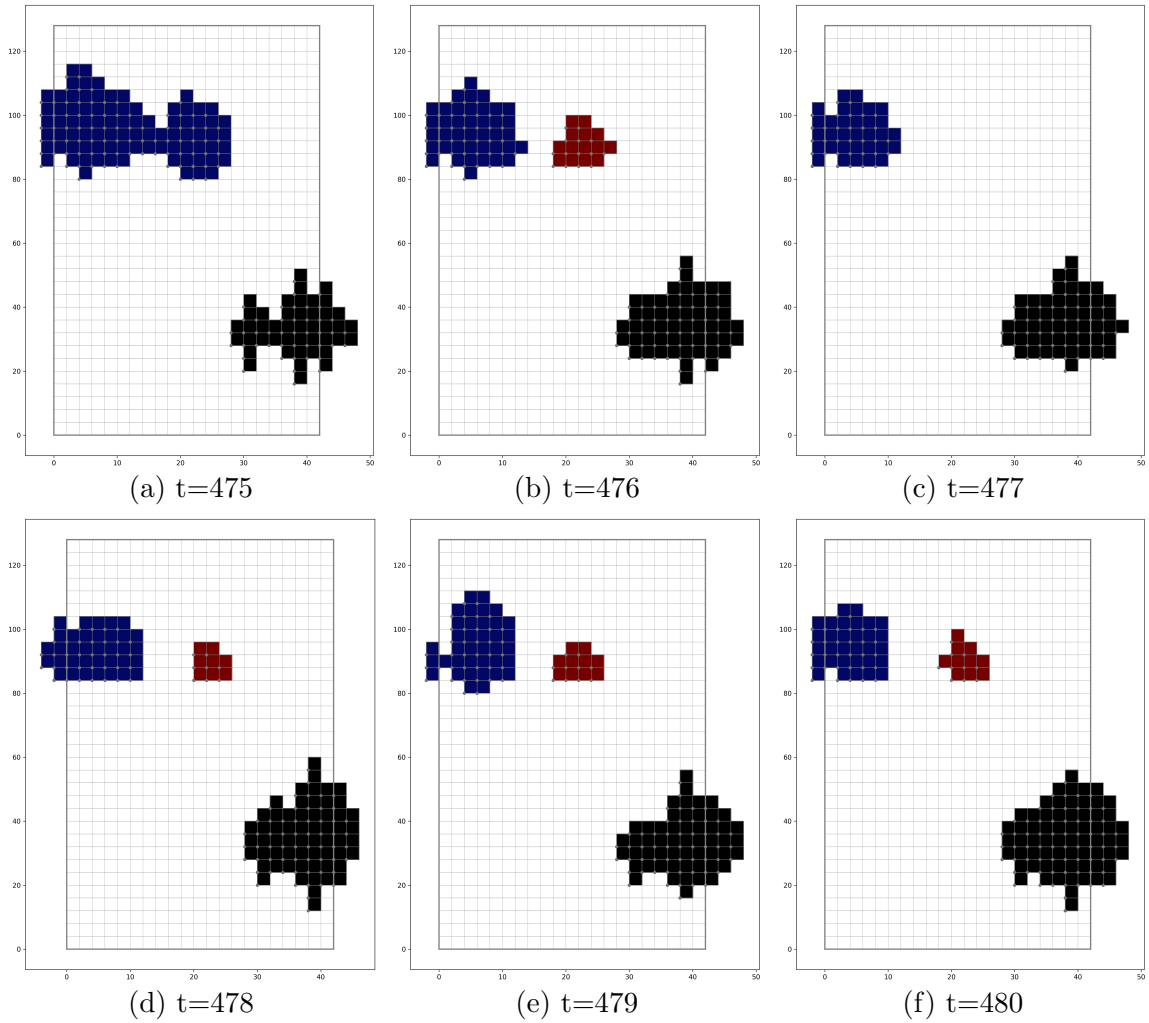
Figure 5.11: Rising particle clusters in three dimensions $t = 230$ to $t = 235$ before joining on periodic boundary



(a) t=230 (b) t=231 (c) t=232

(d) t=233 (e) t=234 (f) t=235

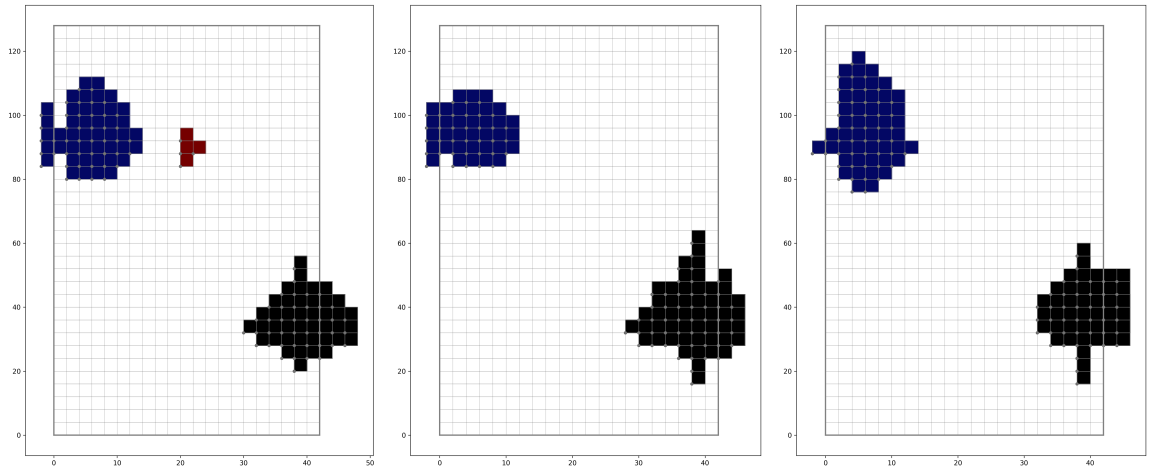
Figure 5.12: Rising particle clusters in three dimensions $t = 230$ to $t = 235$ after joining and being copied on periodic boundary



(a) t=475 (b) t=476 (c) t=477

(d) t=478 (e) t=479 (f) t=480

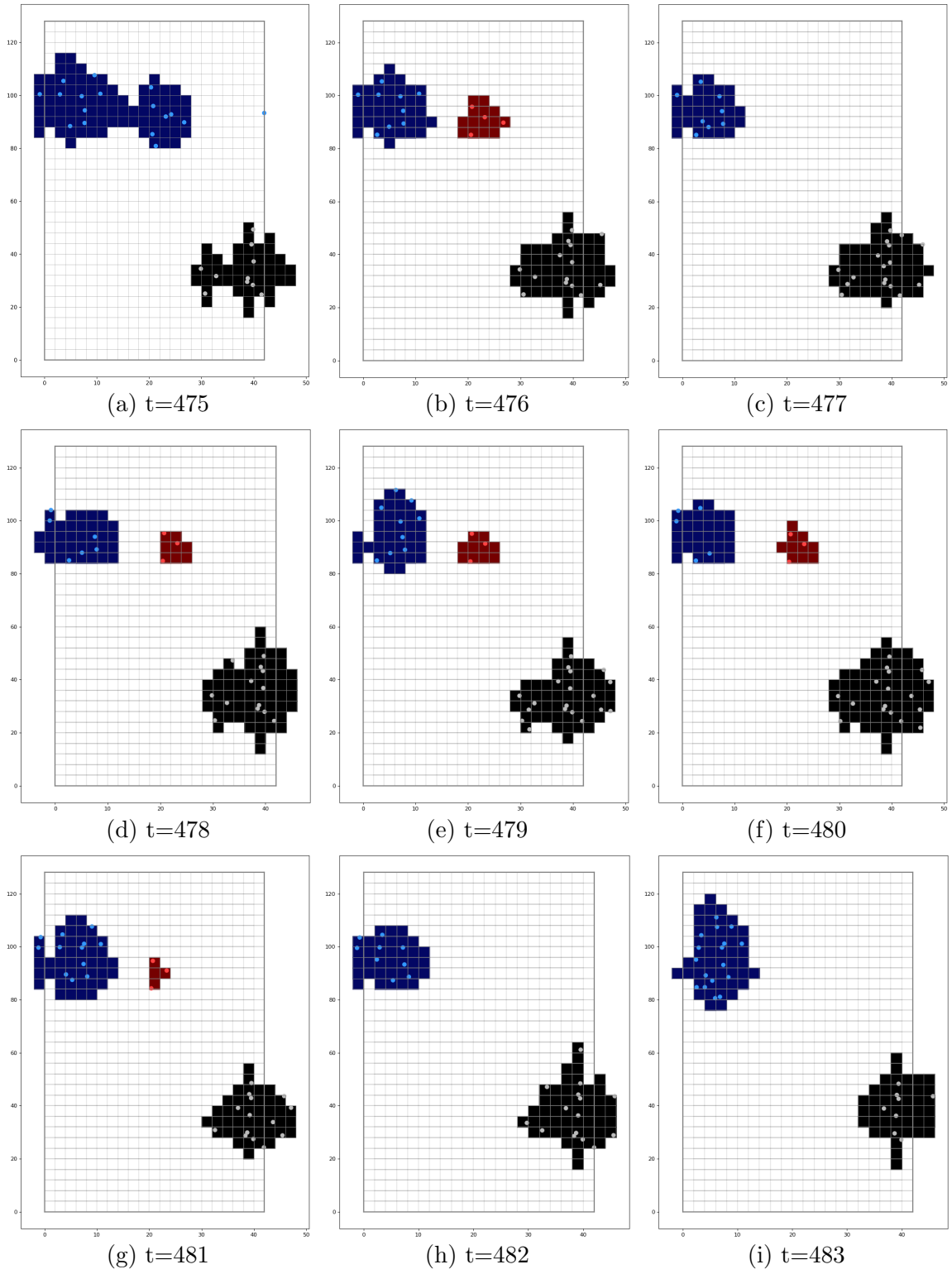
Figure 5.13: Falling cell clusters in xz-plane for time steps $t = 475$ to $t = 480$



(a) t=481 (b) t=482 (c) t=483

Figure 5.14: (continuation of the previous figure) Falling cell clusters in xz-plane for time steps $t = 481$ to $t = 483$

Figures 5.13 and 5.14 show a sequence from $t = 475$ to $t = 483$ of a situation in which multiple clusters are identified. As we can see from these figures, the method described in section 4.6 is able to correctly track the individual clusters over time. Figures 5.15a to 5.15i present the projection of corresponding particle clusters in xz-plane. The colors of the clusters are representative of the final cluster labels. This means cluster $\mathcal{C}_0$ is always colored as black, and clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ are always colored as dark blue and dark red respectively.



(a) t=475 (b) t=476 (c) t=477

(d) t=478 (e) t=479 (f) t=480

(g) t=481 (h) t=482 (i) t=483

Figure 5.15: Falling particle clusters in xz-plane for time steps $t = 475$ to $t = 483$

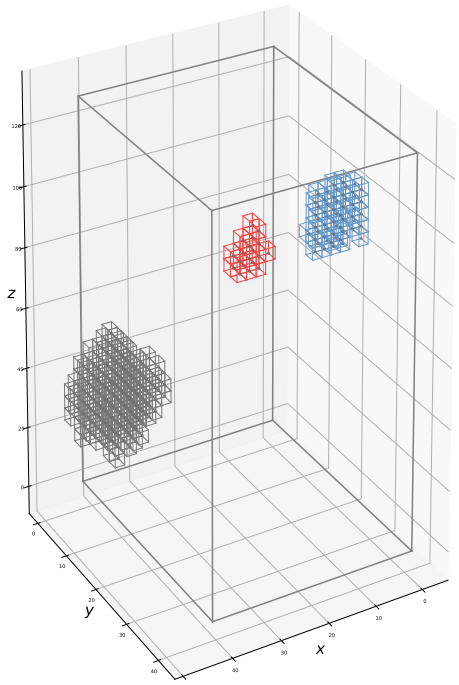
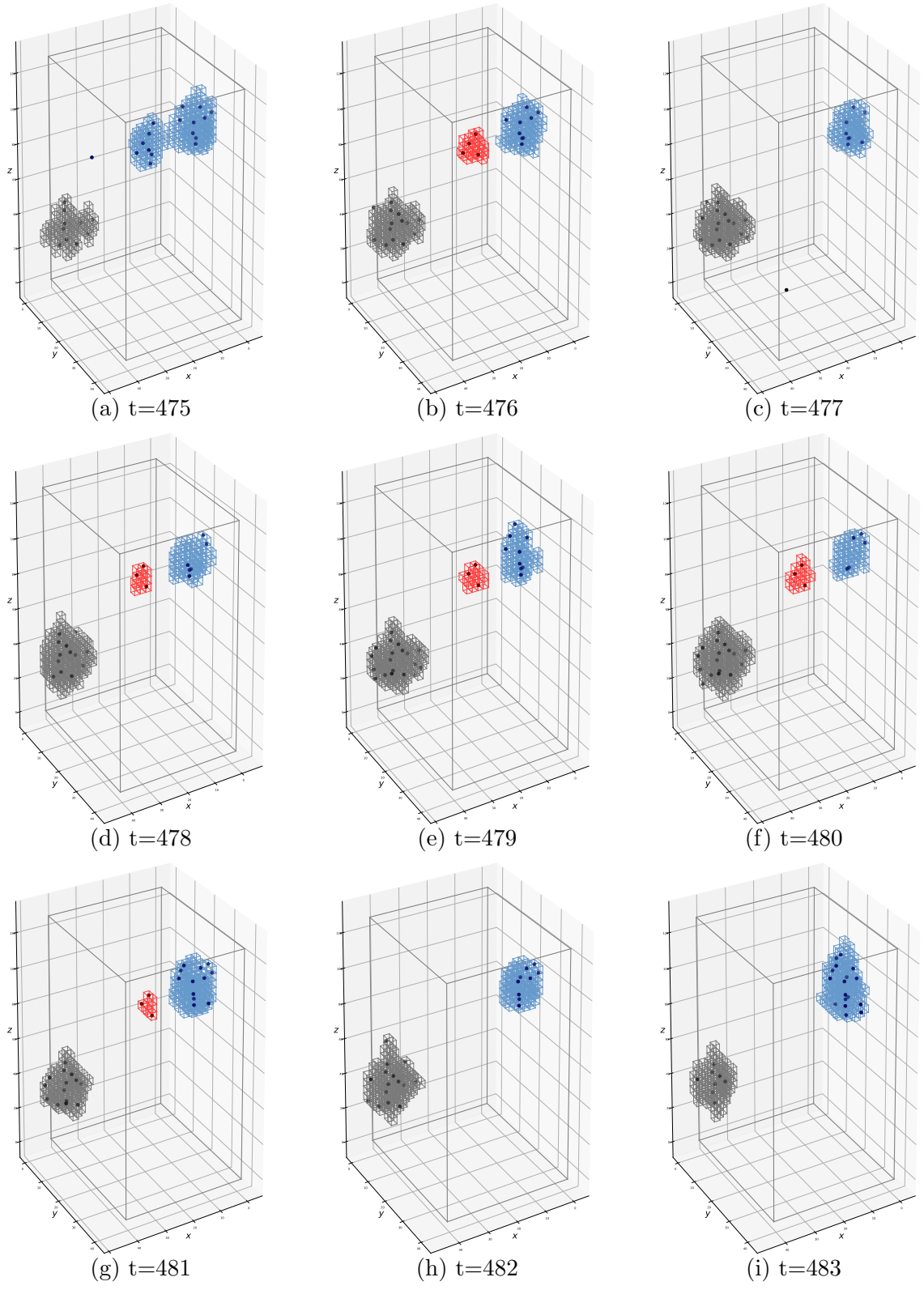


Figure 5.16: Multiple falling cell clusters at time step $t = 480$ in three dimensions

Figure 5.16 shows the the cell clusters of time step $t = 480$. In the following three-dimensional figures, including this figure, the cells with gray edges represent the cell cluster with the final label of 0, i.e., $\mathcal{C}_0$, and the ones with blue and red edge colors represent the cell clusters with the final labels of 1 and 2, i.e., $\mathcal{C}_1$ and $\mathcal{C}_1$ respectively. In addition, the black, blue and red dots represent the corner point for the cell clusters and the particles for the particle clusters, which belong to the clusters the with the label 0,1 and 2 respectively. Figures 5.17 visualize the history of cell clusters of figure 5.13 and of the particle clusters within them in three dimensions.



(a) t=475 (b) t=476 (c) t=477

(d) t=478 (e) t=479 (f) t=480

(g) t=481 (h) t=482 (i) t=483

Figure 5.17: Falling cell clusters and particles clusters within them in three dimensions from $t = 475$ to $t = 483$

5.2 Average Vertical Velocity of Particles in Clusters

The purpose of this work was to identify groups of particles which share a faster-than-average velocity (in modulus). In the previous section, we proved that our method finds and joins clusters and tracks them over time. In this section, we verify that those clusters are indeed constituted by particles with larger velocities than the mean for both falling and rising particles.

The red line in figure 5.18 shows the mean velocities of the falling particles in the clusters identified by our method. The gray line is the mean fall velocity of all the particles in the region R. It is evident that the particles in the clusters fall indeed with a velocity considerably faster than the mean. A similar comparison for the rising particles is shown in figure 5.19. This comparison enables us to conclude that our method detects particle clusters on the basis of our desired property.

A comparison of figures 5.18 and 5.19 shows that the velocity of the falling particles fluctuates more strongly than that of the rising particles. We have already pointed out in connection with figure 5.20 of Chapter 2, reproduced here in figure **??** for convenience, that the mean velocity of the falling particles exhibits a stronger fluctuations than that of the rising particles. We may conclude that this attribute is associated with the stronger fluctuations of the falling particle clusters.

Furthermore, by comparing the average vertical velocity of falling particles in clusters (figure 5.18) and the fluctuations of the standard deviation of the particle vertical velocity of falling particles (figure 5.20a), we see a clear connection especially for the first part of the simulation, for example, the first downward peak in figure 5.18 is responsible for the first upward peak of figure 5.20a. This shows that clustering has a significant contribution to the standard deviation of the falling particles at

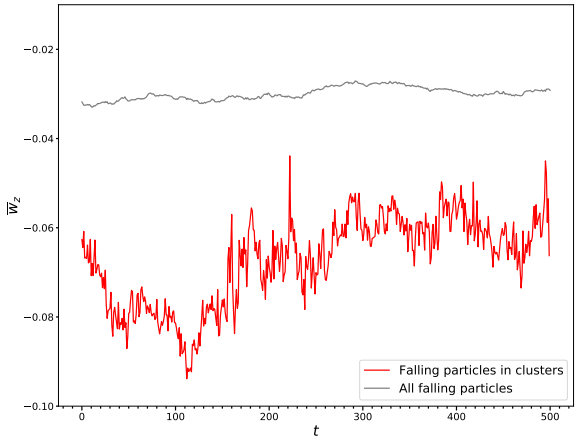


Figure 5.18: Average vertical velocity of falling particles in clusters and all falling particles

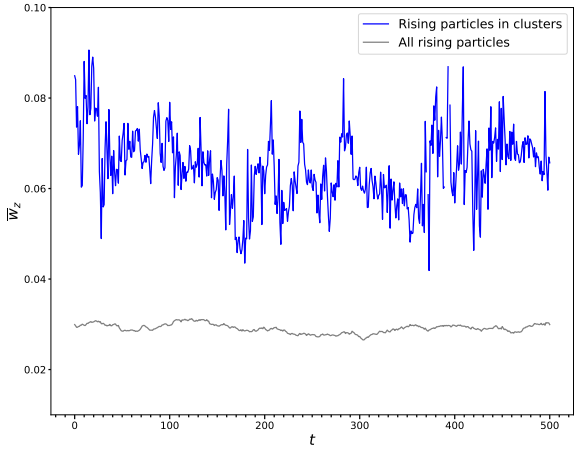
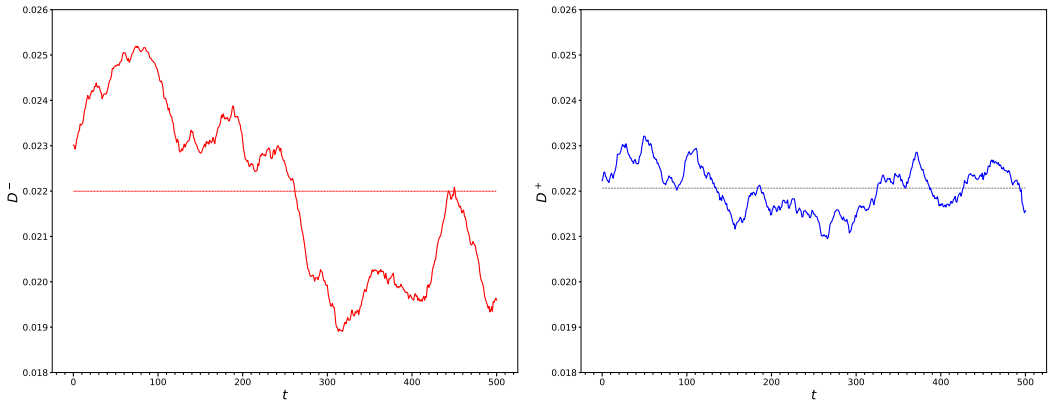


Figure 5.19: Average vertical velocity of rising particles in clusters and all rising particles

least for the first half of the simulation. The same comparison for rising particles (see figures 5.19 and 5.20b) shows some resemblance between clustering and standard deviation of particle vertical velocity. For instance, from $t = 100$ to $t = 180$ we see a descending portion in both figures 5.19 and 5.20b, or around $t = 400$ there is a peak in both figures. These resemblances suggest that clustering can be responsible for the fluctuations of the standard deviation. However, a final confirmation needs further analysis.

As we have described in Chapter 4, not every cell in cell clusters is



(a) (b)

Figure 5.20: Time dependence of the fluctuations of the standard deviation of the particle velocity in the vertical directions with respect to the time-mean values indicated by the horizontal lines for a) falling particles and b) rising particles

interesting. In fact, our method detects the clusters in which their cells are interesting on average. Following this fact, we do not expect that all particles in a particle cluster to be interesting. However, we want the majority of them to be interesting as they form an interesting cluster. To validate our method in this matter, we should check the interesting particles ratio in a cluster. This means how many falling (or rising) particles in a falling (or rising) cluster have the falling (or rising) velocity of $\alpha = 1.5$ standard deviation greater than the average. The ratio of the interesting falling particles in a falling cluster is between 0.3 to 1, and the same ratio for rising particles is between 0.4 and 0.95 for all time steps. This is compatible with the pre-assigned parameter β, i.e., 0.2, which we have used for detecting the interesting masks.

Chapter 6

Summary and Conclusions

In this work we proposed a general method to binarize the region of interest into interesting and non-interesting subregions and to identify the clusters in physical space. The same method can be also applied in other more general spaces. We verified our method with a dataset which is the result of a resolved simulation on equal particles obtained by means of the Physalis method. This method solves the problem of equal particles suspended in an upward fluid flow treating them as moving boundaries for the fluid. In Chapter 2, we briefly described this method. The basic idea is to use analytical solutions to Stokes equations that, due to the no-slip condition, are valid in the immediate neighborhood of the particle surface in the rest frame of the particle. The simulations for the data used in our work are conducted in a triply periodic computational domain with a square cross section of dimension $20a \times 20a$ and a vertical extent of $60a$, in which a is the radius of the particles with 8 mesh lengths per particle radius. By balancing the gravitational forcing in the vertical direction with an imposed upward pressure gradient, the simulations were in effect carried out in a reference frame coincident with the mean vertical particle motion. The total number of particles is 1000 and the the ratio of particle density to

fluid density is 2. We have used the last 501 time steps of these simulation for the purpose of our study.

The initial impulse for this work was to understand whether the simulation produces particles that are in some sense exceptional, i.e., such that they exhibit strong correlation between linear and angular velocities, etc. , or particle groups that share similar values for their attributes. As described in Chapter 3, we have used several methods for this purpose. However, none of these methods produced any interesting results. We then developed a new method, described in Chapter 4, to explore whether the particle groups shared common attributes. This method proved successful and appears likely to be applicable to many problems of this type.

The first step is dividing the region into cells. We then use 20000 masks the corner points and dimensions of which are generated using a uniform random distribution in such a way that each cell of the domain has an approximately equal probability to be covered by a mask. In order to identify the interesting masks, we defined a logical predicate which is True if the at least 20% of the particles in the mask have a vertical velocity which is 1.5 standard deviation from the mean value. Since each mask contains several cells, by counting the number of times that each cell is being covered by an interesting mask, each cell was assigned a value which is equivalent to an intensity value of an image pixel. Using a global threshold, we labeled high intensity cells as interesting and the rest as non-interesting. We have also explained in this chapter how we used an integration of region growing and region merging methods to cluster the interesting cells. In two-dimensional space, we started with the bottom-left corner cell, then going one increment in upward direction, the first interesting cell is the initial seed for growth and is assigned to the first cluster. We examined the above and the right neighbor cells, and if they are interesting, we merge the cells into the same clusters until there is no cell interesting left without cluster label. An important step is to check the intersection of each neighbor set and

each of clusters at every step. The extension to three-dimensional is immediate and has been explained in the end of the chapter. We then explained how to join the clusters that are on the boundary due to periodicity and how to track the clusters in time.

in Chapter 5, we implemented the method on the vertical velocity of particles and their coordinates. We treated the falling and rising particles separately and found two different clusters set for each case. We tested the method for different cases where only a single cluster is formed, multiple clusters are formed on the periodic boundary, and tracking the multiple clusters which their provisional label has changed at some time steps. Therefore, we showed that our method first binarizes the region of the interest in the physical space with small amount of noise, and these regions of interest are accurately identified and tracked over time. At the end, we showed that the average velocity of particles in clusters is larger than the average of all in modulus, therefore, our method identifies clusters of faster or slower particles coorectly. We have also shown that there is some resemblance of standard deviation of vertical velocity with the average vertical velocity of the particles in clusters, which needs further analysis.

In the future, it would be useful to extend the study of the effects of the parameter choices that we have adopted to better characterize the optimal parameter range. Furthermore, other data similar to the ones that we have used are available and it would be interesting to carry out an analysis similar to the one described in this dissertation on them. Of course, the purpose of the method that we have developed ultimately is to understand the physics that gives rise to the clusters that we have identified and to clarify the specific fluid mechanic processes that cause that. However interesting, an analysis of this type lies beyond the scope of our work.

Remaining within the context of fluid mechanics, it would be interesting to examine datasets of a different nature. For example, databases exist of fully developed

homogeneous single phase turbulence (Wu *et al.*, 2020). One could use our method, for example, to identify clusters of vortices in these simulations. Beyond fluid mechanics, our method can be used for identifying the spatial cluster of prevalent disease in a geographical region, or finding the spatial patterns of a crime in a city or segmenting a medical image in order to diagnosis tumors.

References

Boiangiu, A., Boiangiu, C.-A., Ioanitescu, R. & Dragomir, R.-C. 2016 "Voting-based OCR system." *The Proceedings of Journal ISOM* **10**, 470–486.

Boiangiu, C.-A. & Ioanitescu, R. 2013 "Voting-based image segmentation." *The Proceedings of Journal of Information Systems & Operations Management* **7** (2), 211–220.

Costin-Anton, B., Andrei-Iulian, D. & Dan-Cristian, C. 2009 "Binarization for digitization projects using hybrid foreground-reconstruction." *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing, ICCC 2009* pp. 141–144.

Dhage, P., Phegade, M. R. & Shah, S. K. 2015 "Watershed segmentation brain tumor detection." *2015 International Conference on Pervasive Computing: Advance Communication Technology and Application for Society, ICPC 2015* .

Fan, J., Yau, D. K., Elmagarmid, A. K. & Aref, W. G. 2001 "Automatic image segmentation by integrating color-edge extraction and seeded region growing." *IEEE Transactions on Image Processing* **10**, 1454–1466.

Fan, J., Zeng, G., Body, M. & Hacid, M.-S. 2005 "Seeded region growing: an extensive and comparative study." *Pattern recognition letters* **26** (8), 1139–1156.

Feng, J., Cao, Z. & Pi, Y. 2013 "Multiphase sar image segmentation with G^0

-statistical-model-based active contours." *IEEE Transactions on Geoscience and Remote Sensing* **51** (7), 4190–4199.

Gajanayake, G. M., Yapa, R. D. & Hewawithana, B. 2009 "Comparison of standard image segmentation methods for segmentation of brain tumors from 2d mr images." *4th International Conference on Industrial and Information Systems 2009, ICIIS 2009* pp. 301–305.

Gonzalez, R. C. & Woods, R. E. R. E. 2017 *Digital image processing, 4th Ed.*.

Gupta, S. 2018 "A brief survey on image segmentation techniques." *International Journal of Research in Engineering, IT and Social Sciences* **08**.

Hochbaum, D. S. 2009 "Polynomial time algorithms for ratio regions and a variant of normalized cut." *IEEE transactions on pattern analysis and machine intelligence* **32** (5), 889–898.

Hore, S., Chakraborty, S., Chatterjee, S., Dey, N., Ashour, A. S., Chung, L. V. & Le, D. N. 2016 "An integrated interactive technique for image segmentation using stack based seeded region growing and thresholding." *International Journal of Electrical and Computer Engineering* **6**, 2773–2780.

Kang, C. C., Wang, W. J. & Kang, C. H. 2012 "Image segmentation with complicated background by using seeded region growing." *International Journal of Electronics and Communications (AEU)* **66**, 767–771.

Kang, W. X., Yang, Q. Q. & Liang, R. P. 2009 "The comparative research on image segmentation algorithms." *Proceedings of the 1st International Workshop on Education Technology and Computer Science, ETCS 2009* **2**, 703–707.

Kaur, A., Rani, U. & Josan, G. S. 2020 "Modified sauvola binarization for degraded document images." *Engineering Applications of Artificial Intelligence* **92**, 103672.

Kovacevic, N., Lobaugh, N. J., Bronskill, M. J., Levine, B., Feinstein, A. & Black, S. E. 2002 "A robust method for extraction and automatic segmentation of brain images." *Neuroimage* **17** (3), 1087–1100.

Lim, Y. W. & Lee, S. U. 1990 "On the color image segmentation algorithm based on the thresholding and the fuzzy c-means techniques." *Pattern Recognition* **23**, 935–952.

Narendran, P., Kumar, V. K. N. & Somasundaram, K. 2012 "3d brain tumors and internal brain structures segmentation in mr images." *International Journal of Image, Graphics and Signal Processing* **4**, 35–43.

Niblack, W. 1986 *An introduction to digital image processing*. Prentice-Hall International.

Otsu, N. 1979 "A threshold selection method from gray-level histograms." *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-9** (1).

Phromsuwan, U., Sirisisathitkul, C., Sirsisathitkul, Y., Uyyanonvara, B. & Muneesawang, P. 2013 "Application of image processing to determine size distribution of magnetic nanoparticles." *Journal of Magnetics* **18**, 311–316.

Pobleanu, C.-D., Dumitrescu, A., Cristea, S. & Vlasceanu, G. V. 2020 "Imroving the image binarization process by using a weighted-voting scheme." *Journal of Information Systems & Operations Managements* **14.2**.

Preetha, M. M. S. J., Suresh, L. P. & Bosco, M. J. 2012 "Image segmentation using seeded region growing." *2012 International Conference on Computing, Electronics and Electrical Technologies, ICCEET 2012* pp. 576–583.

Pruncu, A., Ghimbas, C., Boeru, R., Neculae, V. & Boiangiu, C.-A. 2018 "Majority image voting binarization." *The Journal of Information Systems & Operations Management* .

Rycroft, C. 2009 "Voro++: A three-dimensional voronoi cell library in c++." *Tech. Rep.*. Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

Saad, N. M., Abu-Bakar, S. A., Muda, S. & Mokji, M. 2010 "Automated segmentation of brain lesion based on diffusion-weighted mri using a split and merge approach." *Proceedings of 2010 IEEE EMBS Conference on Biomedical Engineering and Sciences, IECBES 2010* pp. 475–480.

Saad, N. M., Abu-Bakar, S. A., Muda, S. & Mokji, M. 2011 "Segmentation of brain lesions in diffusion-weighted mri using thresholding technique." *2011 IEEE International Conference on Signal and Image Processing Applications, ICSIPA 2011* pp. 249–254.

Sauvola, J. & Pietikainen, M. 2000 "Adaptive document image binarization." *Pattern Recognition* **33**, 225–236.

Senthilkumaran, N. & Vaithegi, S. 2016 "Image segmentation by using thresholding techniques for medical images." *Computer Science & Engineering: An International Journal* **6** (1), 1–13.

Shaikh, S. H., Maiti, A. K. & Chaki, N. 2013 "A new image binarization method using iterative partitioning." *Machine Vision and Applications* **24**, 337–350.

Shankar, K. C. P. & Shyry, S. P. 2021 "A survey of image pre-processing techniques for medical images." *Journal of Physics: Conference Series* **1911**.

Sierakowski, A. J. & Prosperetti, A. 2016 "Resolved-particle simulation by the physalis method: Enhancements and new capabilities." *Journal of Computational Physics* **309**, 164–184.

Srinivasa, G., Oak, V. S., Garg, S. J., Fickus, M. C. & Kovacevic, J. 2008 "Voting-based active contour segmentation of fmri images of the brain" .

Suzuki, Y., Kawai, H., Ito, K., Aoki, T., Fujio, M., Kaga, Y. & Takahashi, K. 2020 "Hand segmentation for contactless palmprint recognition." *Asian Conference on Pattern Recognition, ACPR 2019* pp. 902–912.

Tripathi, S., Kumar, K., Singh, B. K., Singh, R. P., Singh, B. K. & Singh, R. P. 2012 "Image segmentation: A review." *International Journal of Computer Science and Management Research* **1**.

Wertheimer, M. 1923 "Laws of organization in perceptual forms." *Psycologische Forschung* **4**, 301–305.

Willen, D. P. & Prosperetti, A. 2019 "Resolved simulations of sedimenting suspensions of spheres." *Physical Review Fluids* **4**.

Willen, D. P., Sierakowski, A. J., Zhou, G. & Prosperetti, A. 2017 "Continuity waves in resolved-particle simulations of fluidized beds." *Physical Review Fluids* **2**.

Wu, Z., Zaki, T. A. & Meneveau, C. 2020 "Data compression for turbulence databases using spatiotemporal subsampling and local resimulation." *Physical Review Fluids* **5** (6), 064607.

Xu, G., Bu, Y., Wang, L. & Li, H. 2011 "A color image segmentation algorithm by integrating watershed with automatic seeded region growing and merging." *2011 International Conference on Optical Instruments and Technology: Optoelectronic Imaging and Processing Technology* **8200**, 820018.

Yin, F.-F., Doi, K. & Schmidt, C. J. V. R. A. 1994 "Computerized detection of masses in digital mammograms: Investigation of feature-analysis techniques." *Journal of Digital Imaging* **7**, 18–26.

Zhang, H., Fritts, J. E. & Goldman, S. A. 2008 "Image segmentation evaluation: A survey of unsupervised methods." *Computer Vision and Image*

Understanding **110**, 260–280.

Zhao, Y., Nasrullah, Z. & Li, Z. 2019 "Pyod: A python toolbox for scalable outlier detection." *Journal of Machine Learning Research* **20** (96), 1–7.

Zhu, L., Gao, Y., Yezzi, A. & Tannenbaum, A. 2013 "Automatic segmentation of the left atrium from mr images via variational region growing with a moments-based shape prior." *IEEE transactions on image processing* **22** (12), 5111–5122.