universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Void finder: Towards robust dust cavity detection in astrophysical datasets

verfasst von | submitted by

Amanda de Paula Cristino Hirschl

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 921
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Informatik
Betreut von | Supervisor:	Univ.-Prof. Torsten Möller PhD

Acknowledgements

I hereby express my deepest gratitude to my supervisor, Univ. Prof. Torsten Möller, PhD, for all the meetings, feedback, and invaluable insights that have shaped this thesis.

I also could not have undertaken this journey without the knowledge and expertise of Sebastian Ratzenböck, PhD, whose assistance from the very beginning has been indispensable. I am thankful for all the time and effort you have dedicated to this project.

Finally, I would like to thank my husband, Jakob Hirschl, MSc, whose endless support, encouragement, and patience have been my anchor throughout the journey of writing this thesis.

Abstract

The interstellar medium (ISM), composed of gas and dust, is a dynamic environment constantly shaped by stellar processes. These processes include supernova explosions, that can create low-density regions known as dust voids. Mapping and analyzing these voids is crucial for understanding star formation and the evolution of our galaxy. Recent advances in 3D dust mapping have provided unprecedented resolution, enabling detailed studies of voids. However, the identification of these structures has traditionally relied on visual inspection, which is labor-intensive and prone to errors. This thesis addresses this limitation by introducing the Void-Finder algorithm, an automated method for detecting dust voids in 3D dust distributions.

The algorithm leverages the radial density profile characteristic of supernova-induced voids, identifying regions with a central density minimum surrounded by increasing density that transitions back to the ambient ISM. Designed to overcome the challenges of complex ISM structures, incomplete void boundaries, and the lack of labeled training data, the Void-Finder algorithm eliminates the need for extensive manual inspection and facilitates the discovery of previously unmapped voids. The contributions of this work include the development of an automated solution for void detection in astrophysical 3D dust data, the labeling of voids in an ISM simulation, and a structured assessment of classical and contemporary image segmentation techniques for this task.

The findings presented in this thesis highlight the potential of the Void-Finder algorithm to enhance our understanding of ISM structure and improve efficiency in void detection, by providing a robust automated solution for the problem.

Kurzfassung

Das interstellare Medium (ISM), bestehend aus Gas und Staub, ist ein dynamisches Umfeld, das ständig durch stellare Prozesse, einschließlich Supernova-Explosionen, geprägt wird. Diese Explosionen schaffen Regionen mit niedriger Dichte, die als Staubleeren bekannt sind. Die Kartierung und Analyse dieser Leeren ist entscheidend für das Verständnis der Sternentstehung und der Entwicklung unserer Galaxie. Jüngste Fortschritte in der 3D-Staubkartierung haben eine beispiellose Auflösung ermöglicht, die detaillierte Studien dieser Leeren erlaubt. Die Identifikation dieser Strukturen stützte sich jedoch traditionell auf visuelle Inspektion, was arbeitsintensiv und fehleranfällig ist. Diese Masterarbeit geht auf diese Einschränkung ein, indem sie den Void-Finder-Algorithmus vorstellt, eine automatisierte Methode zur Erkennung von Staubleeren in 3D-Staubverteilungen.

Der Algorithmus nutzt das radiale Dichteprofil, das für durch Supernovae induzierte Leeren charakteristisch ist, und identifiziert Regionen mit einem zentralen Dichteminimum, das von einer zunehmenden Dichte umgeben ist und wieder in das umgebende ISM übergeht. Der Void-Finder-Algorithmus wurde entwickelt, um die Herausforderungen komplexer ISM-Strukturen, unvollständiger Leeren-Grenzen und das Fehlen von gekennzeichneten Trainingsdaten zu überwinden. Er eliminiert die Notwendigkeit einer umfangreichen manuellen Inspektion und erleichtert die Entdeckung zuvor nicht kartierter Leeren. Diese Arbeit stellt die erste automatisierte Lösung zur Erkennung von Staubleeren in astrophysikalischen 3D-Staubdaten vor. Manuell gekennzeichnete Leeren des ISM-Simulationsdatensatzes dienen als „Ground Truth“ für die Validierung des Algorithmus. In Folge wird dieser Datensatz auch für eine strukturierte Bewertung klassischer und moderner Bildsegmentierungstechniken für diese Aufgabe verwendet.

Die in dieser Masterarbeit präsentierten Ergebnisse heben das Potenzial des Void-Finder-Algorithmus hervor, unser Verständnis der ISM-Struktur zu verbessern und die Effizienz bei der Leeren-Detektion zu steigern.

Contents

Acknowledgements i

Abstract iii

Kurzfassung v

List of Tables ix

List of Figures xi

1. Introduction 1

2. Background 5
   - 2.1. Data . . . 5
   - 2.2. Known voids in the dust map . . . 8
     - 2.2.1. Local bubble . . . 8
     - 2.2.2. Per-Tau shell . . . 9
     - 2.2.3. Cepheus Flare shell . . . 11
   - 2.3. Simulations . . . 12

3. Related work 15
   - 3.1. Classical approaches . . . 15
     - 3.1.1. Thresholding . . . 15
     - 3.1.2. Edge-based segmentation . . . 16
     - 3.1.3. Region-based segmentation . . . 18
     - 3.1.4. Deformable models . . . 20
     - 3.1.5. Clustering . . . 22
     - 3.1.6. Graph based segmentation . . . 23
   - 3.2. Deep learning methods . . . 24

4. Void-Finder Algorithm 27
   - 4.1. Seed selection . . . 31
   - 4.2. Selection criteria . . . 31
   - 4.3. Candidate merging . . . 33
   - 4.4. Ellipsoid fitting and density metrics . . . 34

5. Results 39
   - 5.1. Selected methods . . . 39

5.2. Evaluation setup and parameter spaces 41
5.3. Simulation results 44
5.4. Dust map results 47
5.5. Runtime analysis 52

6. Conclusion and future work **53**

Bibliography **55**

A. Data preprocessing **61**
A.1. Selected methods 61
A.1.1. Pre-processing evaluations 62

List of Tables

5.1. Parameter settings 41
5.2. Resulting segmentation of the applied methods for the Per-Tau shell (left) and Cepheus Flare shell (right). Images (a) and (b) display the slices of the resulting segmentation, while images (c) and (d) display the projection of the same regions. 50

List of Figures

2.1. Log-normal distribution of dust density in the dust map. 6
2.2. 2D middle slices of the 3D data. 6
2.3. 3D visualization of the (log) dust density. 7
2.4. Projection of the gradient magnitude of the unprocessed dust map. 8
2.5. Projection of the gradient magnitude of the pre-processed dust map. . . . 8
2.6. 3D view of the Perseus-Taurus region including the Per-Tau shell model (colored sphere) and surrounding dust (in gray). Source: Bialy et al. [8] [Figure 5]. 10
2.7. Central slices and radial profile of the Per-Tau shell. Dashed lines connect the slice locations (top) to the corresponding mean density (bottom). . . . 10
2.8. Cepheus Flare shell in the 3D dust map. Left: the wire frame sphere indicates the void position. Right: dust-only view of the void. The region is a subset of the original map, centered at (-143pc ,248pc ,87pc) with radius of 80 pc. 11
2.9. Central slices and radial profile of the Cepheus Flare shell. Dashed lines connect the slice locations (top) to the corresponding mean density (bottom). 12
2.10. 2D middle slices of the 3D simulation data. From left to right, the images display the slices of Z=256, Y=256 and X=256 respectively. 13
2.11. Simulation labeling process. The image on the left shows the data with supernovae locations (circles), color coded by age (green: recent, red: long time ago). The image on the right shows the labeling of the corresponding slice. 14

3.1. Input image 17
3.2. Prewitt kernels 17
3.3. Sobel kernels 17

4.1. Void-Finder candidate selection. Image (a) illustrates the initial seed grid for a specified region in the dust map containing a void. Image (b) showcases seeds that have been identified by the algorithm as potential void candidates. Image (c) depicts the approximation of some candidate boundaries. 28

4.2. Void-Finder ellipsoid fitting process. Image (a) shows a selected void candidate after the first merging step. Image (b) exhibits the points selected for the ellipse fitting. Image (c) depicts how the distances of these points from the center are updated so that they lie in the densest nearby region. Image (d) shows the ellipse that was fitted based on the selected points. Finally, image (e) displays the initial circle and its center (round point), as well as the fitted ellipse and its center (cross). 29
4.3. Activity diagram of the Void-Finder algorithm. 30
4.4. Visualization of seed generations with the Sobel sequence. On each step 6^2 new seeds are generated, with high sparsity and minimal overlapping. . . . 31
4.5. Mean radial profile and the smoothed gradient for the Per-Tau shell. The vertical line marks the density peak. Gradient points above zero (green) correspond to a density increase in the profile, while points below zero (orange) represent a decrease. 33
4.6. Density peaks in random directions. Image (a) displays the dust map focused on the Cepheus Flare region, including the estimated center and two sample lines used for the ellipsoid fitting step. Lines are drawn in arbitrary directions and the location where the density peaks along the line is used as a fitting point for the ellipsoid. Images (b) and (c) show the density along the lines, highlighting the density peaks. 36
4.7. Ellipsoid density metrics. Image (a) shows how the shell density correlates to the dust in the region, making open areas easier to spot. Image (b) illustrates how the distance between the center of mass of the shell and the ellipsoid center changes when the shell density is balanced (left) or unbalanced (right). 37

5.1. Slices of the simulation data and segmentation results for the different applied methods. 45
5.2. Jaccard scores for simulations dataset. 46
5.3. Overview of void candidates and their relationship to known supernovae events in the simulation dataset. 46
5.4. Void delimitation of ground truth and Void-Finder. Image (a) shows a slice of the ground truth, where two voids can be seen. Image (b) shows the projection of that object through all the slices. Image (c) shows the slice with the corresponding objects found by the Void-Finder algorithm. . 47
5.5. Void candidates and validation voids identified by method for the dust map. 51
5.6. Run times for the selected methods on the dust map. 52

A.1. PSNR variation for Gaussian and median filter 63
A.2. PSNR variation for gradient anisotropic diffusion at different iterations and conductance. 63
A.3. Gradient magnitude projection over different parameters for the Gaussian filter, median filter and anisotropic diffusion. 64

A.4. Volume variation for the segmentation in the Per-Tau Shell region with the different preprocessing filters. 64
A.5. Otsu's segmentation of the Per-Tau Shell region over different parameters for the Gaussian filter, median filter and anisotropic diffusion. 65

1. Introduction

Composed mostly of gas and dust, the interstellar medium (ISM) is everything that fills the space between stars in a galaxy. The gas—comprising hydrogen, helium, and other elements—and the dust, which consists of tiny solid particles, can be found either diffused throughout the ISM or concentrated in denser clouds. Stars are typically formed in these dense clouds of gas and dust. Throughout their lifetimes, stars undergo various stages of nuclear fusion, and once no more energy can be obtained through these processes, they enter the final stage of their existence. If the star is massive, its core collapse can take the form of a supernova explosion, which violently expels its material back into the ISM [30]. This powerful process sweeps up gas and dust which can cause new stars to form [8]. These stellar activities continuously alter the distribution of gas and dust within the ISM. The observation and characterization of this distribution is crucial for helping astronomers to gain better understanding of the star formation process and evolution of the Milky Way galaxy.

An intriguing feature of the ISM is the presence of dust voids, also known as dust cavities, bubbles, or shells. They are regions of low dust density that are surrounded by higher dust density, believed to have been formed through one or multiple supernovae explosions. One of the best-known examples of such voids is the Local Bubble, which surrounds the Sun [73]. Apart from the voids already known in the literature, some of which include the Per-Tau shell [8], the Cepheus bubble [60], the Cepheus Flare Shell [21], the Perseus shells [3] and the OBP-B1 bubble [16], many voids are believed to remain unmapped. The effect of supernovae explosions in the formation of dust voids is also supported by numerical simulations of the ISM [64].

Astronomers typically use a multifaceted approach to locate and accurately map these dust voids. This involves analyzing various observables within specific areas of interest, such as the positions of stars, dust densities, magnetic fields, spectroscopic data, velocities, among others. However, the process of integrating data from different telescopes and satellites can be challenging due to variations in scale and accuracy among the sources [74]. Recent advancements in data collection have improved this scenario, enabling the achievement of unprecedented levels of precision in mapping the three-dimensional distribution of dust near our Sun [36]. This high-resolution mapping has made it possible for astrophysicists to study the three-dimensional characteristics of supernovae voids, a feat never before accessible in such detail [74, 8].

Nevertheless, the advent of new data comes with its own set of challenges. Up to this point, the identification of these voids relied on visual inspections of the data. Yet, the projections of 3D data onto 2D screens can often be deceptive, and outcomes may diverge between different domain experts. Given the novelty of this topic, the development of an automated method for void detection remains largely unexplored.

Finding dust voids on 3D density data is a task that can naturally be framed as an image segmentation problem. But the highly complex distribution of the ISM poses unique challenges that classical segmentation techniques struggle to overcome. The ISM, in its constantly evolving state, exhibits substantial disparities in local densities, influenced by a variety of processes beyond just supernovae. This leads to voids that are not neatly enclosed, having low density 'holes' in its surrounding dust envelope, and existing across various density levels. The lack of labeled data also imposes a constraint for supervised methods. While existing numeric simulations of the ISM incorporate supernovae information that could be used for labeling, they cannot yet accurately replicate the complexity and intricacy of the observed ISM distribution. As a result, these simulations cannot be used 'off the shelf' for training models intended for real datasets and instead require sophisticated post-processing to bridge the gap between the simulation domain and the data domain.

The Void-Finder algorithm is proposed in this thesis as a tailored solution for this task. It analyzes the 3D dust distribution to identify potential dust void candidates. This eliminates the need for tedious visual inspection of large 3D maps in the search for new dust voids, improving the efficiency of the process. The algorithm is specifically designed to examine the dust density in a radial pattern around a central point. Supernovae explosions sweep up the constituents of the ISM, pushing them radially outwards. Therefore, the algorithm anticipates that the density within the void formed by the explosion will be at its lowest around the void's central point, gradually increasing as one moves away from this center. Beyond the radius expansion influenced by the explosion, the density begins to decrease again, marking the boundary where the ISM returns to its unaffected state, a pattern exploited by the Void-Finder algorithm in its search for voids.

The contributions of this work are:

- First automated solution for void detection in 3D volumes within an astrophysical context: The Void-Finder algorithm is a tailored solution for identifying dust voids in 3D dust distributions, capable of detecting voids even when they are not fully enclosed by dust, a challenge classical image segmentation approaches struggle to overcome.

- Data labeling: For the evaluation of the presented methods, simulations of the ISM dust distribution were labeled and can be used as ground truth of voids for the dataset.

- Structured evaluation of image segmentation tools: various image segmentation methods are assessed for the task of identifying dust voids, highlighting their effectiveness and limitations.

The thesis is structured as follows: Chapter 2 provides more information on the domain background and available datasets. Chapter 3 provides a list of related works in the field of image segmentation. Chapter 4 introduces the Void-Finder algorithm. Chapter 5 provides a comparison of established image segmentation methods against the Void-

Finder algorithm. Finally, Chapter 6 discusses the results, potential improvements, and limitations of the proposed algorithm.

2. Background

This chapter provides an overview of the available data and simulations that describe the dust distributions in the interstellar medium, as well as the patterns that can be found on the density distribution of existing voids.

2.1. Data

Dust and gas are the main constituents of the interstellar medium (ISM). Although the majority of the ISM is composed of gas rather than dust, the dust can be measured through a phenomenon called *extinction*. In this process, dust particles can either absorb or scatter light, impacting our ability to observe celestial bodies at different wavelengths in the electromagnetic spectrum [31]. This makes the dust presence and distribution easier to detect compared to gas. Given that gas and dust are roughly equally distributed throughout the ISM [28], mapping the location of dust through extinction enables us to also infer the distribution of gas. Interstellar dust is also abundant in star-forming regions, being a crucial part of the life cycle of stars [37].

In recent decades, there have been numerous efforts to create three-dimensional maps of interstellar dust in our galaxy [41, 13, 49, 20, 37, 36]. These maps rely on measurements of dust extinction as a way to approximate the density distribution of dust in the ISM. Until recently, the primary emphasis of dust maps was to offer a broad perspective on dust over large scales, with some even revealing the structure of the spiral arms of our galaxy [36]. However, this comes at the cost of lower spatial resolutions, with each voxel corresponding to several cubic parsecs[1], scales that are not sufficiently detailed for supernovae void detection.

This scenario has been significantly improved thanks to the dust map introduced by Leike et al. [36]. The authors achieved an unprecedented level of detail in their reconstruction of nearby dust clouds by leveraging the new data releases from the European Space Agency's GAIA mission, and integrating observational data from the ALLWISE, PANSTARRS, and 2MASS surveys to create a unified catalog with stellar parameters. In constructing the map, the authors employed variational inference and Gaussian processes to model the density of dust extinction. The dust map reconstructs the nearest dust clouds up to a distance of approximately 400 pc, with a resolution of 1 pc [36]. This is relatively small, considering that the Milky Way has a diameter of approximately 26.8 kpc (kiloparsecs) [18].

[1] A parsec (pc) is an astronomical unit of distance, which is approximately 3.26 light years (the distance light travels in one year).

Due to its remarkable resolution, this map is well-suited for detecting supernovae voids. The 3D dust map has dimensions of 740 pc x 740 pc x 540 pc, with each voxel corresponding to a distance of 1 pc per side. The value of each voxel is the estimated dust extinction density in that volume [36]. Overall, the dust distribution follows a log-normal distribution, as shown in Figure 2.1. For easier interpretability, the data has been scaled to a range of 0 to 100, since the logarithmic transformation (applied using NumPy) maps the original dust density values (ranging from 6.77×10^6 to 3.98) to negative numbers.

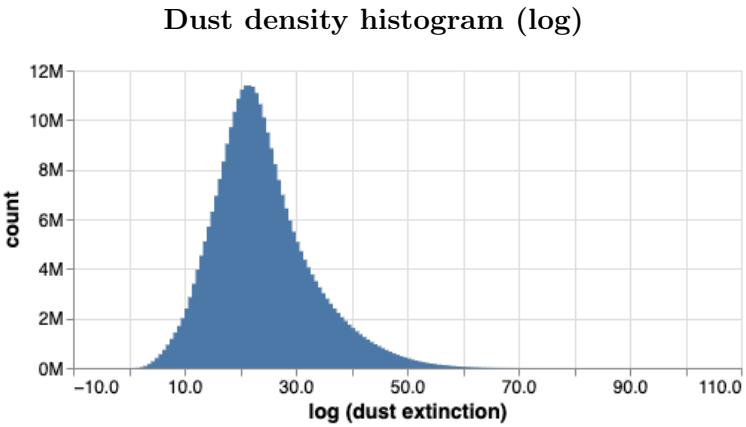


Figure 2.1.: Log-normal distribution of dust density in the dust map.

Throughout the next chapters, both 2D and 3D visualizations are used to represent the dust map. The 2D slice visualizations refer to the central slices of the 3D data: the XY slice is taken at Z=270, the YZ slice at X=370, and the XZ slice at Y=370. The 2D projections follow a similar convention, but instead of showing a single slice, they display the sum of all values along each axis. The data slices can be seen in Figure 2.2, and a 3D visualization of the dust map can be seen in Figure 2.3. For reference, the Sun would be located in the center of the map, at coordinates (0,0,0).

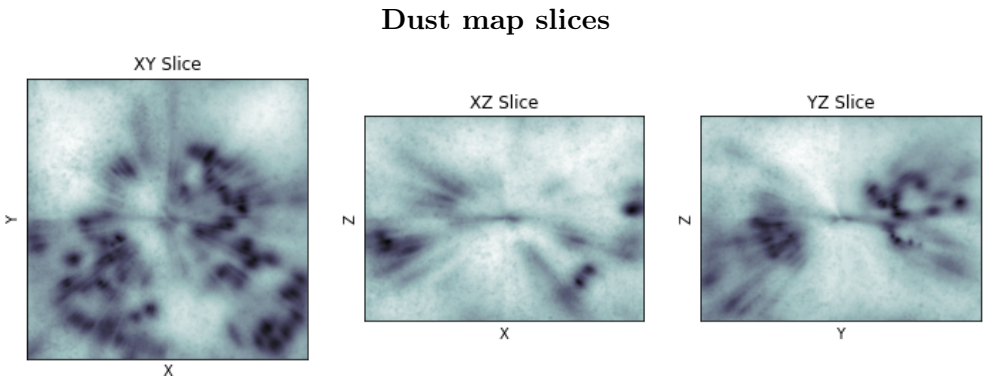


Figure 2.2.: 2D middle slices of the 3D data.

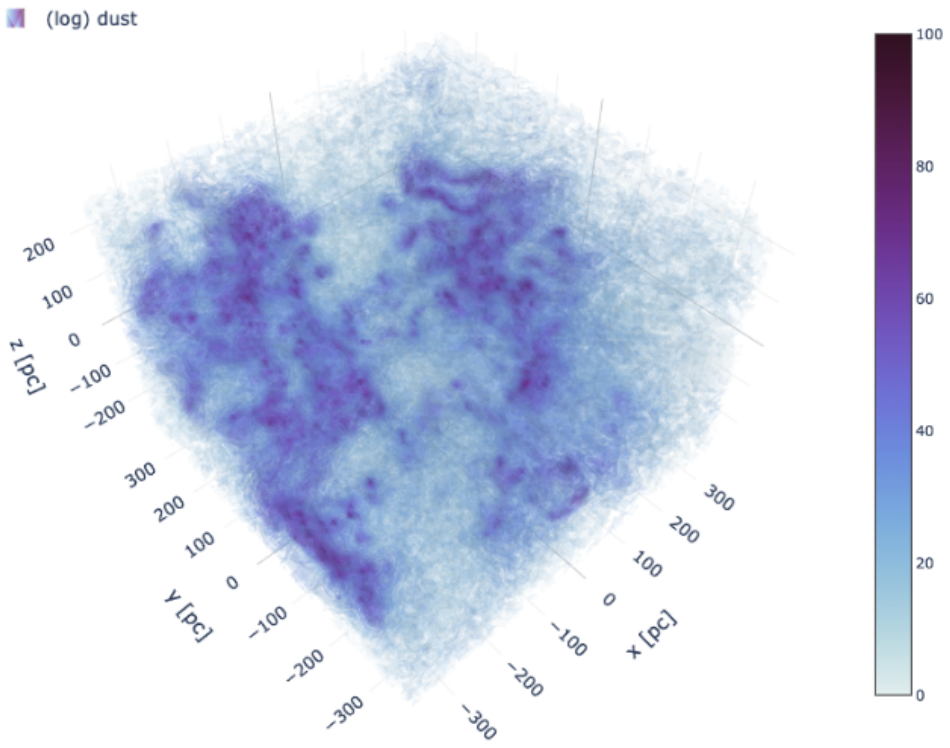


Figure 2.3.: 3D visualization of the (log) dust density.

For parallelization purposes, the authors of the dust map have split the volume into eight octants that were independently reconstructed and combined into the full map using an interpolation scheme [36]. This has led to some artifacts at the borders where these octants meet. By examining the gradient magnitude projection of the map, as shown in Figure 2.4, it is possible to identify sharp changes in the dust density, which highlight boundaries and transitions within the distribution. This reveals that each quadrant exhibits varying levels of intensity compared to one another. These variations highlight the need for pre-processing before the map can be used for automated void detection, as this may impact the quality of the results.

To enhance the octants coherence, pre-processing filters such as Gaussian, median filter, and anisotropic diffusion were evaluated. The median filter, which smooths voxel values by calculating the median of neighboring voxels, was ultimately chosen for pre-processing, as it offered a good balance between smoothing octant transitions and preserving data details. The Gaussian filter, although effective for general smoothing, was found to overly blur fine details. The anisotropic diffusion filter, while capable of preserving edges, did not blend the octant borders as well as the other methods. Figure 2.5 shows the gradient magnitude projection of the pre-processed map, where the *MedianImageFilter* from the

Insight Toolkit (ITK) [70] was applied with radius=4. Further details on the evaluation of these pre-processing filters, along with the metrics used, are provided in Appendix A.

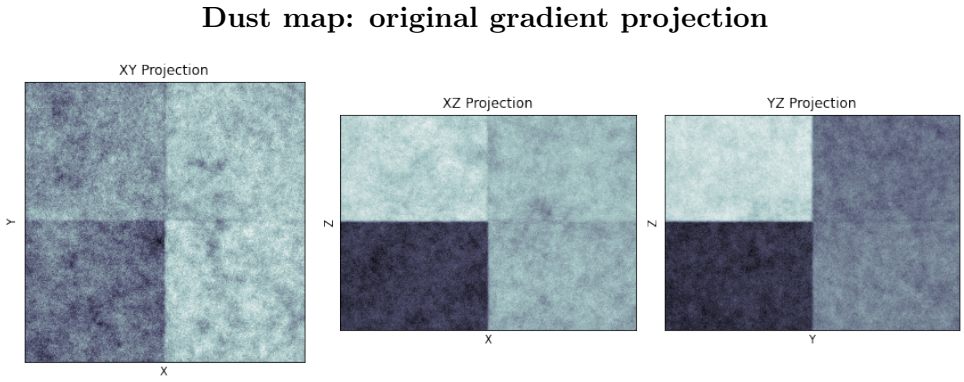


Figure 2.4.: Projection of the gradient magnitude of the unprocessed dust map.

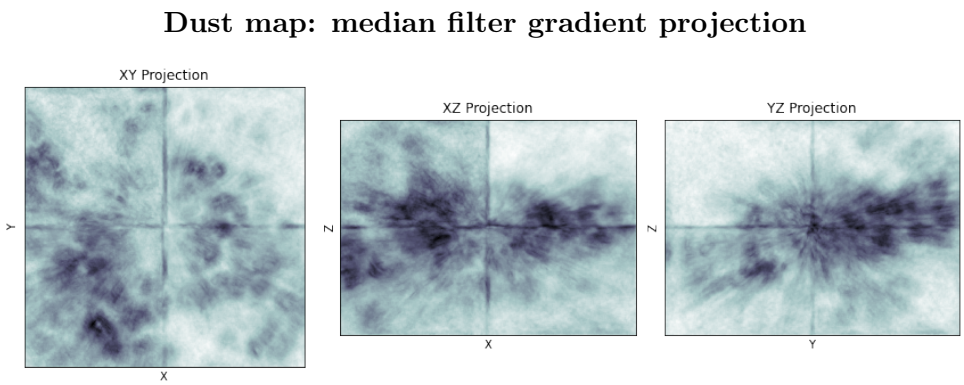


Figure 2.5.: Projection of the gradient magnitude of the pre-processed dust map.

2.2. Known voids in the dust map

While there are various voids documented in the literature, the majority either fall outside the scope of the dust map or are significantly smaller in size when compared to the map's scale. This section offers a summary of the main voids that are within the map range and resolution. It is essential for any automated algorithm to be capable of recognizing these voids, as they serve as ground truth, being the primary qualitative indicator for evaluating the results.

2.2.1. Local bubble

Although the existence of the Local Bubble has been known for decades [15], only recently the astrophysical community began to gain a deeper understanding on its extent and

relationship to nearby star formation. Zucker et al. [73] have reported that nearly all star-forming complexes near the sun lie on the surface of the Local Bubble, supporting the notion that the material ejected from supernova explosions serves now as the foundation for new star formations.

Despite the fact that the Local Bubble is present in the map and that Zucker et al. [73] made available a model of its shape, it will not be used for evaluations on this work. This is primarily due to its vast size, which occupies a significant portion of the map and complicates localized evaluations. Additionally, Leike et al. [36] point out that there is a non-negligible amount of dust in the local bubble. There is also evidence that the bubble is no longer a fully closed surface but instead has evolved into an open "Galactic chimney" [73, 65]. Focusing on smaller voids will allow for a better evaluation of the techniques available for void identification.

2.2.2. Per-Tau shell

The dust map by Leike et al. [36] was also used by Bialy et al. [8] in the discovery of the Per-Tau shell, a void that the authors describe as having a "near-spherical geometry, prominent dust-free cavity, and large extent" [8] [p.7]. The void is estimated to be centered at coordinates (-190,65,-84) in the dust map and has a radius of 78 pc [8].

The Per-Tau shell holds the distinction of being the first supernova void to be identified through the meticulous visual analysis of a 3D dust map. Therefore, it serves as a crucial benchmark for validating methods aimed at identifying voids within the ISM. It is essential to emphasize that any new void candidates are only recognized within the context of the 3D map. Further validation requires supplementary observations of HI, Hα, X-rays, among others, all of which serve as crucial supportive evidence of the supernova scenario.

Within the map, the Per-Tau shell is a challenging entity to pinpoint due to its persistently open regions, which defy conventional image segmentation techniques. This can be seen in Figure 2.6, that shows the 3D view of the Perseus-Taurus region from different angles. Although dust resides on the void's surface, the surface is not entirely enclosed by it at the given iso-surface level.

When analysing the radial profile of the Per-Tau shell, a pattern characteristic of a supernova event becomes evident. In such an event, the exploding star expels gas and dust from a central point, sweeping up material that forms an approximately spherical over-density around it. Afterwards the density falls off to meet the local ISM average. As a result, the density is lower near the center, rises as it approaches the void's boundary, and decreases again beyond that point. This is seen in Figure 2.7, where the top three images are the central slices of the 3D region around the Per-Tau shell, and the figure below shows the mean radial profile from the void's center outwards, with corresponding dashed lines connecting slice locations to radial mean density in the chart.

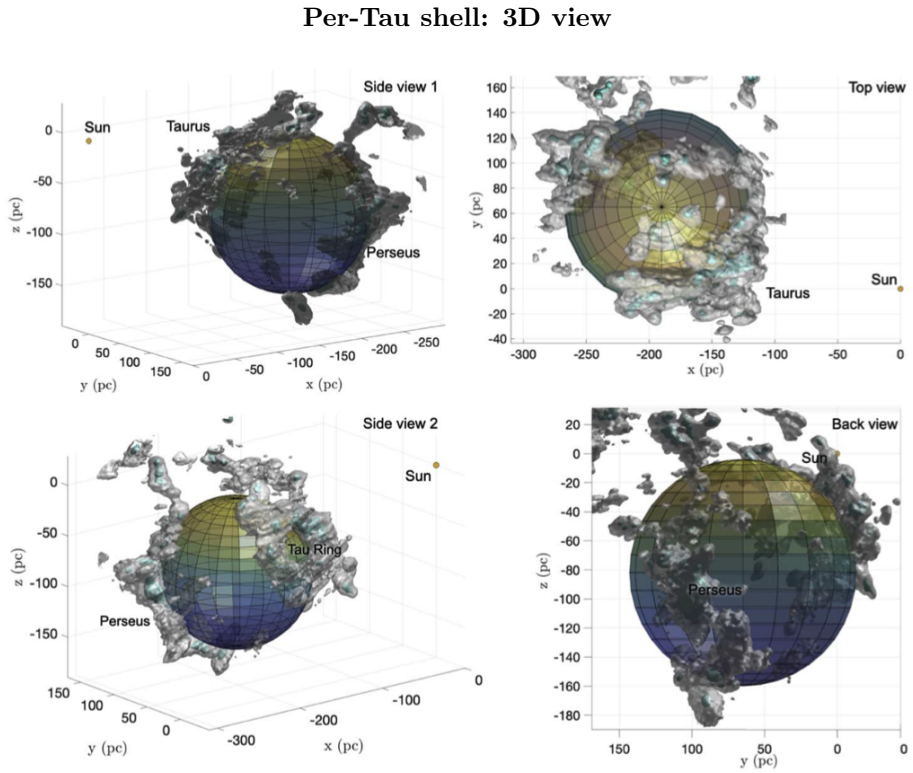


Figure 2.6.: 3D view of the Perseus-Taurus region including the Per-Tau shell model (colored sphere) and surrounding dust (in gray). Source: Bialy et al. [8] [Figure 5].

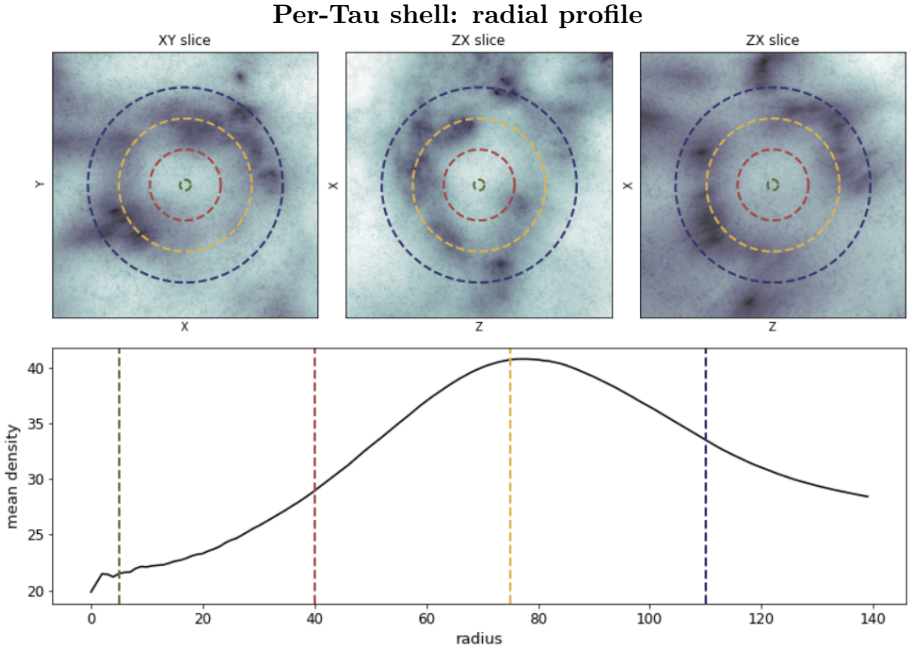


Figure 2.7.: Central slices and radial profile of the Per-Tau shell. Dashed lines connect the slice locations (top) to the corresponding mean density (bottom).

2.2.3. Cepheus Flare shell

The Cepheus Flare region is a substantial complex of gas and dust. Observations of CO velocity dispersions, HI, soft X-Ray and radio lead Grenier et al. [21] to suggest the presence of a void between Cepheus and Cassiopeia, along with the interpretation that it was caused by an old supernova remnant. Olano et al. [46] have later performed a detailed analysis of the kinematics and spatial distribution of the interstellar gas in the region and agreed with the interpretation of Grenier et al.[21].

An intriguing element in the Cepheus Flare shell is a ring shaped structure that is called the 'North Celestial Pole Loop'. The analysis of Olano et al. [46] sheds light on the origins of this gas ridge, suggesting it may be the result of material expelled through a chimney effect from the Cepheus Flare shell, indicating a shell that is open at the top [46].

The shell is positioned at a distance of 300 parsecs from the sun and has a radius of about 50 parsecs [46]. Significantly, this void falls well within the range of the dust map's coverage, making it another compelling candidate for the evaluation of automated void detection algorithms.

Just like the Per-Tau shell, the Cepheus Flare shell seems to also not be fully enclosed by a thick layer of dust, which agrees with simulations that random low density regions in the ISM can create tunnels through which much of the energy can escape. This can be seen in Figure 2.8. The radial mean density profile from the center outwards is shown in Figure 2.9, and also presents the pattern of a curve that peaks at the borders of the void.

Cepheus Flare shell: 3D view

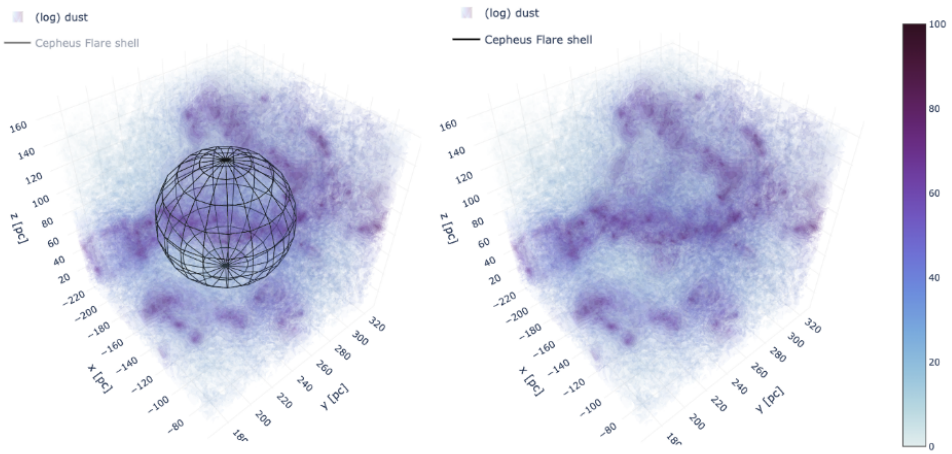


Figure 2.8.: Cepheus Flare shell in the 3D dust map. Left: the wire frame sphere indicates the void position. Right: dust-only view of the void. The region is a subset of the original map, centered at (-143pc ,248pc ,87pc) with radius of 80 pc.

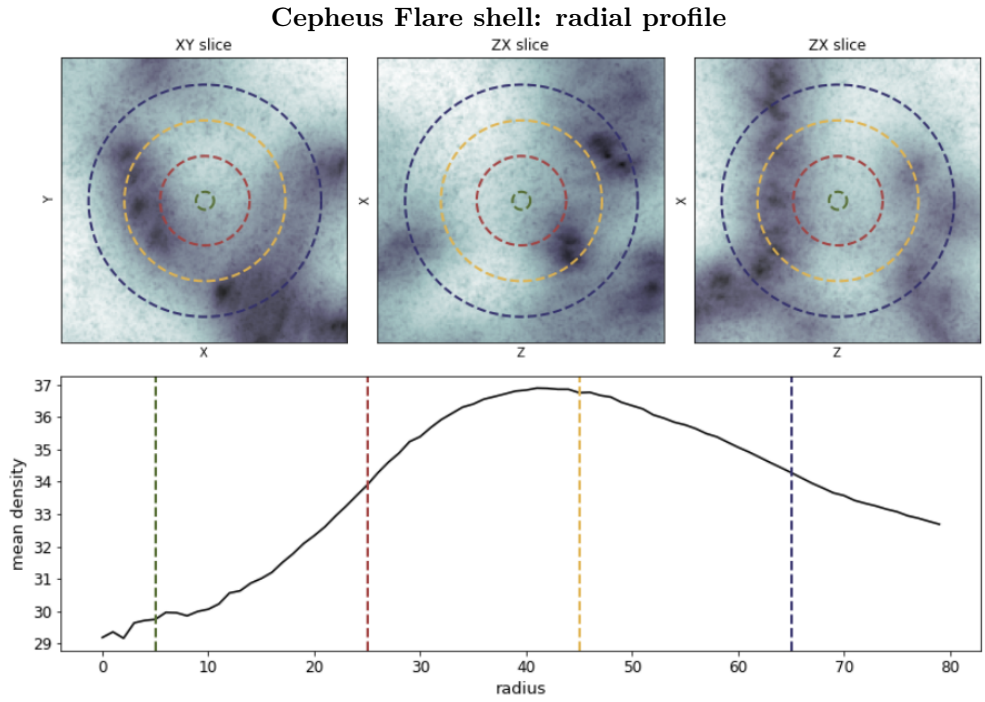


Figure 2.9.: Central slices and radial profile of the Cepheus Flare shell. Dashed lines connect the slice locations (top) to the corresponding mean density (bottom).

2.3. Simulations

The limited number of known voids within the dust map poses a challenge for testing and evaluating potential automated void detection methods. As an alternative for this problem, there exist simulations that reproduce the impact of supernovae explosions in the ISM. The SILCC (SImulating the LifeCycle of molecular Clouds) [64] project provides many of such simulations. The project aims at unraveling the characteristics of the interstellar medium (ISM) on smaller scales and how it relates to the evolution of galaxies, which include supernovae explosions at different rates in high-density regions, at clustered or random locations over time [64].

To provide a more comprehensive understanding of the methods presented in this work, one of the simulations was selected to be included as an additional validation dataset. The main advantage of the simulation data against the real dust map is the availability of the locations of the supernovae explosions, as well as the point in time in which they occurred. Figure 2.10 shows the center slices of the simulation.

Although the information about the supernovae is crucial for evaluating dust voids in the simulations, it is not sufficient by itself to determine the shape and characteristics of the voids. To establish a ground truth for this data, it was necessary to manually delineate and label the voids in the simulation. The locations of the supernovae were used to distinguish voids that were caused by explosions. Additionally, the age of the supernovae

influenced the labeling process, as the shapes of voids created by recent supernovae were much more evident in the simulation than those formed by older events. Figure 2.11 shows the labeling process with the available supernovae information.

Despite the considerable time and effort invested in mapping the simulation voids, the results are not perfect. Given the size of the volume data (512x512x512) and time constraints, manually labeling each voxel was not feasible. The labeling process utilized the tool *ilastik* [6], which assisted by filling in gaps in regions that were not manually labeled. While this method yielded overall good results, it does not compare to the accuracy and detail of complete manual labeling. Another challenge was the difficulty in defining many areas, as some edges were clear and sharp, while others were indistinct. Additionally, the influence of multiple supernova explosions contributed to complex void shapes, posing challenges for both the tool and the manual labeling process.

With the labeled voids serving as ground truth for this synthetic data, it is now possible to evaluate and compare the performance of various segmentation techniques, ensuring a robust assessment of their capabilities in identifying dust voids in the interstellar medium. Having established the necessity of simulations to generate reliable void locations, the next chapter will discuss the methodology for void detection within these 3D dust maps. By utilizing the three-dimensional dust map as a 3D gray-scale image and employing image segmentation techniques adept at distinguishing the density variations, dust voids may be identified and analyzed. This sets the stage for exploring the image segmentation methods employed in this thesis.

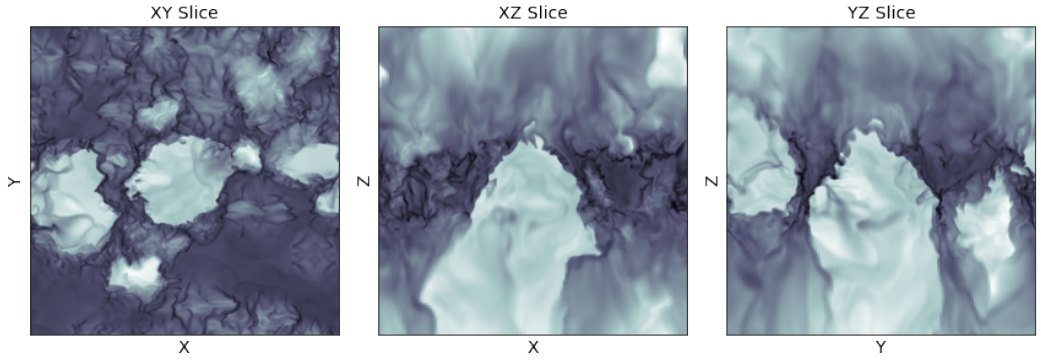


Figure 2.10.: 2D middle slices of the 3D simulation data. From left to right, the images display the slices of Z=256, Y=256 and X=256 respectively.

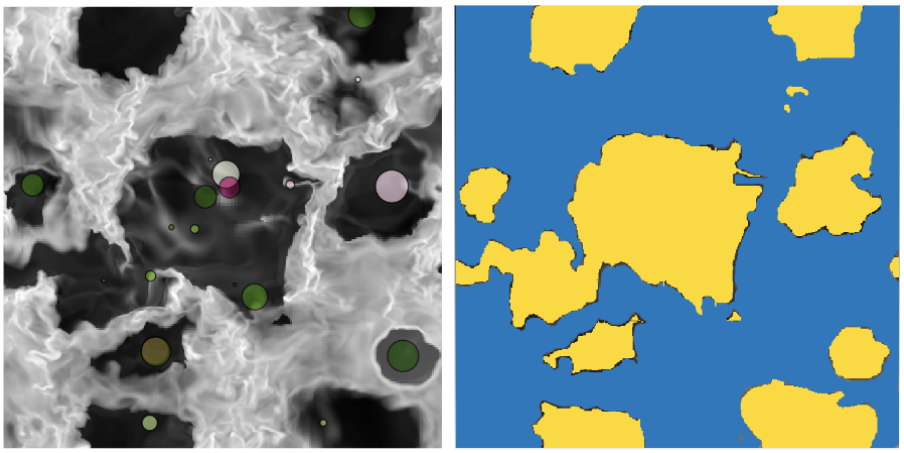


Figure 2.11.: Simulation labeling process. The image on the left shows the data with supernovae locations (circles), color coded by age (green: recent, red: long time ago). The image on the right shows the labeling of the corresponding slice.

3. Related work

As described in the previous chapter, the three-dimensional dust map serves as an approximation of the dust density distribution in a region of our galaxy. This data can be interpreted as a 3D grayscale image, where high intensities correspond to regions with dense dust concentrations, and low intensities indicate sparse dust regions.

The task of identifying voids in such 3D density images involves detecting regions of low density enclosed by higher-density areas. Given the voxelized structure of the data and the need to partition it meaningfully, image segmentation methods provide a natural and effective framework for tackling this problem.

Over the years, a variety of image segmentation techniques have been proposed in the literature. This chapter presents a concise overview of these methods, with a focus on the diverse strategies employed in the field. While the notation predominantly references 2D images (pixels) for clarity, these techniques are inherently extendable to 3D data (voxels), making them directly applicable to the analysis of dust maps.

3.1. Classical approaches

3.1.1. Thresholding

Thresholding is one of the most fundamental concepts in image segmentation. It refers to the categorization of pixels in an image into different groups based on their intensity, color, or other properties. Most thresholding methods aim to separate objects of interest, or foreground, from the background of an image.

In their simplest variants, thresholding methods convert a grayscale image into a binary image by selecting a threshold value that separates the pixels into two distinct classes. The choice of threshold value directly impacts the quality and accuracy of the segmentation results. Selecting an optimal threshold involves considering the specific characteristics of the image and the segmentation objectives. One common technique is to use a fixed global threshold, where a single threshold value is applied uniformly across the entire image. The class of each pixel (x, y) is defined according to the gray value $f(x, y)$ in relation to the threshold value θ, as defined below [66].

$$g(x,y) = \begin{cases} 1 \text{ if } f(x,y) > \theta \\ 0 \text{ otherwise} \end{cases}$$

The threshold value can be defined based on different information from the input image, including the image's histogram shape, entropy minimization or maximization, as well as spatial and statistical data [52].

Among the global threshold techniques, Otsu's threshold is one of the most studied and benchmarked over the years. It is known for its effectiveness in a wide range of image types, including both grayscale and color images. It works by maximizing the inter-class variance or minimizing the intra-class variance in an image that is assumed to have a bimodal intensity distribution. Since it automatically determines an optimal threshold for image segmentation, it does not require any manual user input. It is also computationally efficient and straightforward to implement, serving as a good baseline of a fast and simple method [66, 53].

Variations of the original Otsu method include approaches that use 2D [68, 38] and 3D [57, 7] histograms, which are able to capture more of the spatial information of the pixels, instead of only the gray level. Despite the improvement in the results and robustness to noise and uneven illumination, a higher computational complexity is the trade-off to be considered for these approaches.

Niblack [45] and Sauvola [50] thresholds are also well known methods among local adaptive threshold techniques. In this modality, unlike the global threshold counterpart, a different threshold value is applied to each pixel of the image. Niblack's method [45] selects the threshold values based on the local mean $m(x, y)$ and standard deviation $s(x, y)$ in the area delimited by a defined window size w. A parameter k controls the effect of the standard deviation in the calculation:

$$T(x, y) = m(x, y) - k \cdot s(x, y).$$

One shortcoming of Niblack's threshold is the effect that smooth areas with little noise can have on the segmentation due to the low variance [66]. Sauvola's method [50] aims to solve this issue with the addition of a parameter R, a normalization parameter for the standard deviation, reflecting its dynamic range. The threshold T is calculated as follows:

$$T(x, y) = m(x, y) \cdot \left[1 + k \cdot \left(\frac{s(x, y)}{R} - 1\right)\right].$$

The selection of the value R is not critical for the performance of the algorithm as long as the range of $s(x, y)/R$ is between 0 and 1 for most of the pixels. In practice, the segmentation yields good results when the window size approaches the size of the objects in the image [66].

From the perspective of detecting voids in the dust map, it is known that the distribution of dust can vary in the different regions where supernovae events occur. It is, therefore, expected that global thresholds do not fully characterize the complex void structures. In this scenario, local adaptive threshold methods are likely to yield better segmentation, specially when considering that low and high dust density are relative terms defined locally, and not globally.

3.1.2. Edge-based segmentation

Another common approach to image segmentation is edge-based segmentation, which focuses on identifying and locating boundaries or edges within an image. These edges

represent sharp discontinuities in intensity values and play a crucial role in recognizing, disclosing, and segmenting various image artifacts [53].

Edge segmentation filters can be categorized into two primary groups: first-order filters and second-order filters. These two categories serve distinct purposes in detecting edges within an image. First-order filters, exemplified by operators like Prewitt, Sobel, and Canny [10], are designed to identify broader or thicker edges by assessing how intensity values change across the entire image. In contrast, second-order filters, which include Laplacian and zero-crossing operators, excel at pinpointing finer and more subtle edges present within the image [53].

First-order filters use the first (partial) derivatives of the gray-scale input image f, which are calculated for each pixel at coordinates (x, y) in the image. With that, the gradient and magnitude can be defined for arbitrary directions. The main difference between Prewitt and Sobel operators lies on the kernel used to convolve the image and enhance the potential edges. While the Prewitt mask can be defined by the difference of first and third row or column with even coefficients, the Sobel operator includes more weight in the center part of the mask [56]. A representation of 2D horizontal and vertical kernels for these methods can be seen in Figures 3.2 and 3.3, respectively.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Figure 3.1.: Input image

-1	0	1
-1	0	1
-1	0	1

-1	-1	-1
0	0	0
1	1	1

Figure 3.2.: Prewitt kernels

-1	0	1
-2	0	2
-1	0	1

-1	-2	-1
0	0	0
1	2	1

Figure 3.3.: Sobel kernels

The gradient (∇) of a 2D image f can be denoted as follows [17]:

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The image gradient for the Prewitt is presented here, with the zs corresponding to the intensity values of the image [71], as in Figure 3.1.

$$g_x = \frac{\partial f(x,y)}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$g_y = \frac{\partial f(x,y)}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

Sobel operator works similarly, with a change in coefficients [71]:

$$g_x = \frac{\partial f(x,y)}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$
$$g_y = \frac{\partial f(x,y)}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

While these algorithms are efficient and can produce good results in clear images, they struggle with noise and edge discontinuity. The Canny operator tries to improve these effects by incorporating Gaussian smoothing for noise reduction, suppression of pixels of the gradient when the intensity is lower than the pixels with the same and opposite gradient direction, dual thresholding for strong and weak edges and selecting the final edges with hysteresis thresholding [10]. These enhancements make the Canny operator a more versatile and robust edge detection technique, widely used in computer vision applications.

On the other hand, second-order edge operators focus on capturing the rate of change of intensity by using the second derivatives. An example is the Marr-Hildreth [40] edge detector. In this method, the input image is convolved with a Gaussian (G) filter to reduce noise and provide a smoother representation of the image. Then a Laplacian (∇^2) kernel is applied to the smoothed image to approximate the second derivatives, which correspond to edges in the original image. Alternatively, those steps can be merged into one with the use of a Laplacian of Gaussian (LoG) kernel directly. Since the Laplacian kernel is invariant to rotation, it does not require multiple kernels for different directions. The final step involves identifying zero-crossings in the image. Zero-crossings occur where the sign of at least two opposing pixels of the Laplacian changes. Although the algorithm can perform better than the first-order operators, it is more computationally intensive due to more complex set of operations that are performed. It may also struggle in detecting very thin edges, since the smoothing filter can lead to a loss of information about extremely narrow transitions in intensity [40, 56]. The algorithm convolves an input image with a LoG kernel

$$g(x, y) = \left[\nabla^2 G(x, y)\right] \star f(x, y)$$

and then determines the edges based on the zero crossings of $g(x, y)$ [17].

Many of the presented methods were improved to a variety of applications since their inception. In recent years, many of these operators are combined with deep learning approaches. A comprehensive overview of current techniques can be found in the work of Sun et al. [59].

Since edge detection relies on the identification of sharp intensity or color changes, which are often indicative of object boundaries, an image that does not have well-defined edges, contains noise, or exhibits gradual transitions in intensity will likely produce inaccurate or incomplete segmentation [2]. In the context of void detection in the dust map, edge detection methods could be useful to detect dust clouds that surround potential voids. However, it is unlikely that the voids themselves can be segmented effectively. As mentioned in Chapter 2, the dust surrounding the voids is not isotropically distributed and can contain holes (e.g., chimneys), making edge detection practically unusable.

3.1.3. Region-based segmentation

Region-based segmentation methods focus on finding homogeneous intensity regions in a given image. The fundamental approach for such methods consists of adding pixels of

an image X into regions R, following a similarity criterion L. The following rules apply [19]:

$$X = \cup_{i=1}^{N} R_i$$
$$R_i \cap R_j = 0 \;\; \forall_{i,j} = 1, 2, ..., N$$
$$L(R_i) = TRUE \; for \; i = 1, 2, ..., N$$
$$L(R_i \cup R_j) = FALSE \; for \; \forall_{i,j} = 1, 2, .., N; i \neq j$$

Here, N is the total number of distinct regions of the image X. R_i is i^{th} the region, and $L(.)$ is a logical predicate that contains the similarity criteria of the region, which must be fulfilled by all pixels belonging to the region. The predicate L should fail for the union of two distinct regions since that would result in a heterogeneous region [19].

Region growing and region split and merge are two common approaches for this class of methods [53]. Region growing methods iteratively form homogeneous regions within an image based on a set similarity criteria. The process begins with the selection of seed points which act as starting locations for segmentation. A similarity criterion or function is established to measure the similarity between pixels or small regions and the current growing region. This criterion typically considers properties such as intensity, color, or texture. The algorithm initializes the growing region with the seed points and structures data to keep track of the included pixels or regions. Subsequently, the algorithm iteratively expands the region by adding neighboring pixels or regions that meet the similarity criteria. This process persists until a stopping criterion is met, which could be based on factors like region size, a predetermined number of iterations, or a threshold for similarity [1]. This type of method allows for segmentation of regions of varying shapes and sizes, which is often an advantage. Some of the common challenges faced by region growing algorithms is the need for manual interaction to select suitable seed regions and sensitivity to noise [19, 2].

One extensively used region growing method is the confidence connected region growing algorithm [2], available in the Insight Toolkit (ITK) [70]. The criterion used in this method is based on statistics of the regions, which are defined by the seeds. It works by calculating the mean and standard deviation for all pixel values in the region. Then, a confidence interval is set based on a 'multiplier' parameter, which is multiplied by the standard deviation to establish a range around the mean. Neighboring pixels that fall inside this range are included in the region. Once no more pixels fulfill the criterion, the algorithm iteratively recalculates mean and standard deviation for the regions until a specified number of iterations are met [72].

A variation of this method is an adaptive region growing algorithm [62], that uses a maximum curvature strategy based on the confidence connected region growing algorithm with an increasing and relaxing factor f, and has shown better segmentation for tumor segmentation on PET scans in comparison to the ITK confidence connected algorithm [62]. Despite the better results, as with many other variations, the improvements often come at the cost of high computational complexity [2].

Region split and merge methods are similar to region growing methods in that they aim to partition an image based on a homogeneity criterion. The process initiates by

subdividing the image into regions based on the homogeneity of intensities within specific areas. This subdivision is often done with quad-trees, where each node gives rise to four descendants. The splitting process may result in the creation of multiple closely resembling regions. Consequently, a merging step can be employed, evaluating adjacent regions for similarity and facilitating the formation of larger, more cohesive regions. This merging process iterates until the entire image is segmented into regions that align with the desired homogeneity criteria [17]. This process is efficient and can be more robust to noise than region growing [53].

Watershed is another popular region-based technique. It involves treating the gradient of an image as a topographic surface. Markers, either user-defined or obtained through other segmentation methods, initiate a flooding process on this surface. As the flooding progresses, watershed lines emerge where flooding fronts meet, dividing the image into segments. These segments correspond to catchment basins, and regions separated by watershed lines are labeled accordingly. While effective in delineating objects with clear boundaries, the watershed method may lead to over-segmentation [33, 19]. In practice, watershed is often used in combination with other methods to improve results and minimize over-segmentation.

For void detection, region-based methods face several challenges, including sensitivity to seed selection and a requirement for relatively homogeneous regions, which is at odds with the intricate and varied distribution of the interstellar medium. Additionally, these methods may struggle to detect open voids, as the boundaries can be ill-defined. On the other hand, their capacity to segment irregular shapes can be a notable advantage, making the technique worth exploring.

3.1.4. Deformable models

Deformable models are a powerful class of techniques for identifying and delineating objects within images. These models are particularly valuable when dealing with complex and variable shapes in 3D images. They work by creating a flexible, connected representation of an object or region of interest within an image, leveraging prior knowledge like location, size, shape, and orientation of the object to iteratively adjust and refine their shape until it accurately matches the boundaries of the target object based on energy minimization. Deformable models can be broadly categorized into parametric and geometric models [19].

Parametric deformable models

Parametric deformable models, often referred to as active contour models or snakes [32], employ parametrically defined curves that can adapt to object boundaries, evolving through an energy minimization process balanced by internal and external forces. Internal forces are responsible for maintaining smoothness and continuity of the model's shape, while external forces, derived from the image data, drive the model to adjust its position towards the local minima and close to the object boundaries [32].

The energy E associated with the curve $\mathcal{C}$ is given by

$$E(\mathcal{C}) = \alpha \int_0^1 |\mathcal{C}'(q)|^2 dq + \beta \int_0^1 |\mathcal{C}''(q)|^2 dq - \lambda \int_0^1 |\nabla I(\mathcal{C}(q))| dq.$$

Here, the parametric curve $C(q)$ is used to represent the boundary of an object in the image, where q is a normalized parameter that varies from 0 to 1. The parameter q traces the contour, such that $C(q)$ specifies the spatial position of the contour at the parameter value q. The energy functional $E(C)$ for the active contour model consists of three main terms. The first term, $\alpha \int_0^1 |C'(q)|^2 dq$, enforces the smoothness of the curve by minimizing the magnitude of its first derivative, thereby ensuring the contour does not exhibit sharp discontinuities. The second term, $\beta \int_0^1 |C''(q)|^2 dq$, penalizes high curvature by minimizing the magnitude of the second derivative, which helps maintain the rigidity of the contour and prevents abrupt bending. The third term, $-\lambda \int_0^1 |\nabla I(C(q))| dq$, attracts the contour to edges in the image by maximizing the gradient of the image intensity I at the contour's location $C(q)$. The coefficients α and β are real positive terms governing the internal energy, while λ denotes a real positive term regulating the external energy. The task of solving the snake problem entails identifying, for a specified set of constants α, β and λ, the curve $\mathcal{C}$ that minimizes the energy E [11, 32].

While this model offers a flexible and controllable framework for object boundary delineation, it struggles when faced with concavities, sharp corners, non-homogeneous areas and topology changes [2]. Furthermore, it requires an accurate initial placement near the target object's boundary, with improper initialization leading to convergence issues [69].

Numerous model variations have been proposed to overcome these drawbacks. The Balloon model defines an additional force that inflates the curve so that it behaves like a balloon, stopping only at strong edges. This solves the initialization issue, allowing the contours to be initialized further away from the true contours [14]. The Gradient Vector Flow (GVF) method introduces a diffusion of gradient vectors as a new external force, resulting in a larger capture range and solving the issue of contours fitting into concave regions [69]. The Boundary Vector Field (BVF) model [58] tries to improve the computational performance of the GVF method by introducing a new external force calculated by an interpolation scheme based on the object and image boundaries [58].

To overcome the issues with topology changes, a topological adaptable snakes model was proposed by McInerney and Terzopoulos [42]. This model leverages simplicial domain decomposition and split and merge techniques, where a simplicial grid is superposed over the image to approximate the contour of the object and reparameterize the snake iteratively. This allows for changes in topology and complex shape detection for 2D images, albeit at the expense of higher computational complexity [42].

Geometric deformable models

Geometric deformable models, also known as level sets, differ from parametric models in that they do not rely on a specific parametric representation of the object boundary. Instead, these models represent shapes implicitly as the zero level set of a higher-dimensional function, which evolves over time according to differential equations influenced by intrinsic

geometric properties. Unlike parametric models, geometric deformable models do not rely on a specific boundary parameterization, making them ideal for handling complex shapes and topological changes, thus overcoming the primary limitations of parametric models in 3D image segmentation [19, 43].

The level set method evolves based on a closed surface established by a moving interface $\Gamma(t)$ that expands to the target boundary. The interface is defined as [43]:

$$\begin{cases} \phi(t, \mathbf{p}) > 0 & \text{for } \mathbf{p} \text{ inside } \Gamma(t) \\ \phi(t, \mathbf{p}) < 0 & \text{for } \mathbf{p} \text{ outside } \Gamma(t) \\ \phi(t, \mathbf{p}) = 0 & \text{for } \mathbf{p} \text{ on } \Gamma(t) \end{cases}$$

Here, Γ is the evolving boundary denoted at time t by the zero level set $\Gamma(t) = \{\mathbf{p} \mid \phi(t, \mathbf{p}) = 0\}$ for the level set $\phi(t, \mathbf{p})$. The vector $\mathbf{p}$ represents coordinates of the pixels or voxels of the image. This level set evolution can be defined by the differential equation [43]:

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = 0$$

with F representing the speed function of the curve and ∇ the spatial gradient operator. The speed function is similar to the energy function of parametric deformable models [43].

While most deformable model algorithms depend on edge information, there are cases where edges can be disregarded, as in the algorithm developed by Chan and Vese [12]. The proposed model employs a stopping term with Mumford–Shah segmentation techniques, allowing it to identify boundaries that are not defined by gradients or are very smooth. It also does not require the initial curve to be placed around the objects to be detected, allowing a more flexible placement in the image [12].

An interesting variant of the Chan-Vese model leverages morphological operations to improve segmentation efficiency and reduce computational costs, which is beneficial for processing large 3D datasets. The morphological Chan-Vese method evolves the contours through morphological operations instead of partial differential equations (PDEs). The main advantage is that the morphological operations do not face the same numerical stability issues that are commonly found in PDEs, and are computationally faster [39].

Methods that do not depend on edge information and are robust against noise are especially valuable for identifying voids in dust maps. This is due to the intrinsic characteristics of supernovae voids, which often lack clear boundaries and exist within complex, irregular distributions shaped by supernova explosions in the interstellar medium (ISM).

3.1.5. Clustering

Clustering is an unsupervised learning technique used to group objects into clusters based on shared characteristics. The primary objective is to maximize similarity within clusters while minimizing similarity between them. Unlike supervised learning methods, clustering does not require predefined classes or training data; instead, it relies on iterative processes.

This technique is widely employed in data mining for analyzing large datasets, extracting meaningful insights, and enhancing storage and retrieval efficiency.

In the context of 3D image segmentation, clustering often relies on features derived from the spatial and intensity distribution of the data. Common features include spatial coordinates (x, y, z), intensity or density values at each voxel, gradient magnitudes to highlight edges or transitions, and other descriptors that capture local patterns in the data. By representing the image as a point cloud or voxel grid, clustering algorithms can segment the data into regions of interest based on feature similarity, revealing structures such as voids without requiring labeled training data.

A fundamental and widely used clustering algorithm is k-means, known for its simplicity and efficiency. The algorithm begins by selecting k mean values, corresponding to k clusters. In each iteration, data points are assigned to the cluster with the closest mean, after which new mean values are recalculated for each cluster. This process continues until the mean values stabilize, resulting in a final partitioning of the data into k clusters. However, despite its simplicity and efficiency, k-means may not always yield optimal results, particularly in the presence of noise. Additionally, it requires prior knowledge of the number of clusters, a task that can be both time-consuming and labor-intensive [63].

To address some of the limitations of the traditional k-means algorithm, k-means++ was developed. This variant improves upon k-means by strategically selecting initial cluster centers rather than choosing them randomly. It employs a probability distribution that favors points that contribute more to the overall potential, ensuring that the initial centers are well-spaced and more representative of the data. This approach reduces the likelihood of poor clustering results, leading to better accuracy and faster convergence. Nonetheless, k-means++ still carries the risk of converging to a local optimum and incurs a higher computational cost during initialization due to the need to calculate distances between points [5].

Clustering algorithms can be effective for identifying dust voids since they can group voxels based on similarities in density and other characteristics. This approach allows for the analysis of spatial and intensity values, as well as additional information such as intensity gradients and local density statistics. However, incorporating these extra features can increase the data's dimensionality and lead to higher computational costs.

3.1.6. Graph based segmentation

Graph-based methods in image segmentation utilize graph theory to partition an image into meaningful regions or objects. In these methods, an image is represented as a graph, where each pixel or group of pixels corresponds to a node, and edges are established between neighboring nodes to represent their relationships. The weights assigned to these edges are critical, as they indicate the cost of segmenting the pixels into different categories. The key concept in these methods is to formulate segmentation as a graph-cut problem, with the objective of finding an optimal cut that separates objects of interest from the background. Common approaches include minimum cuts and normalized cuts [2].

The minimum cut approach is based on the concept of minimizing the cost of a cut.

While a cut partitions the graph's nodes into disjoint subsets, the cost is determined by the sum of the weights of the edges that are severed. In this framework, edge weights reflect the likelihood of a pixel belonging to either the object or the background. The minimum cut can be efficiently computed using combinatorial optimization algorithms such as the max-flow/min-cut theorem. This theorem states that the maximum flow in a network corresponds to the minimum cut. By iteratively adjusting the flow along the edges, these algorithms identify the optimal partition. This results in a segmentation that minimizes the boundary cost between the object and the background, effectively delineating the object of interest [9].

Another technique in graph-based segmentation is the Normalized Cut (N-Cut) method. The N-Cut evaluates partition quality by balancing the dissimilarity between segments with the similarity within each segment. By normalizing the cut value relative to the sizes of the segments, the N-Cut method avoids biases that could arise from merely minimizing the number of edges cut. This approach formulates the problem as a generalized eigenvalue problem, enabling efficient computation of optimal partitions. The N-Cut method is particularly effective for capturing an image's global structure, focusing on the overall layout rather than just local features. However, despite its ability to produce coherent and meaningful segmentations, the N-Cut method can be sensitive to noise, the choice of similarity measures, and the parameters used to define the graph. Additionally, the computational complexity can be significant, especially for large images, as solving the eigenvalue problem can be resource-intensive [54].

For identifying dust voids, graph-based methods offer the potential to model complex relationships between voxels, allowing for effective segmentation of irregular and non-uniform structures. However, their computational complexity are a significant limitation in their practical application.

3.2. Deep learning methods

In recent years, deep learning methods have seen a big surge in popularity. Their capacity to automatically learn intricate features from data has made them the go-to choice, particularly with Convolutional Neural Networks (CNNs) leading the way. Deep learning automates feature learning, adapts to diverse tasks, and can achieve top-tier performance with large datasets.

In supervised learning, data is crucial. In essence, the input data is the foundation upon which deep learning models are built, and high-quality, well-labeled data that accurately represents the problem domain is essential for training robust models and achieving optimal performance. Many techniques, such as data augmentation, transfer learning, active learning, and synthetic data generation, can mitigate the issues of limited data. However, they often require domain knowledge or at least a few instances of labeled data [61]. Despite having labeled the simulations dataset, the fact that it is only one instance of data with ground truth, and knowing how its characteristics can greatly vary from the original data, the requirements of such methods do not optimally align with the specifics of the case at hand.

An alternative approach that does not rely on labeled data is deep unsupervised learning. While deep learning methods have become popular, the research landscape has primarily concentrated on supervised learning, with relatively limited attention directed toward unsupervised methods.

Xia and Kulis proposes the W-Net architecture, which employs two fully convolutional networks (FCNs) in an autoencoder architecture. The first FCN converts input images into flexible soft segmentations, while the second FCN reconstructs the images from these segmentations. The model's goals are to minimize the image-reconstruction error and impose soft constraints on segmentation. In a post-processing step the authors use conditional random field smoothing and hierarchical segmentation to enhance the final segmentations [67].

One of the first works using convolutional neural networks (CNN) in a fully unsupervised manner for image segmentation was proposed by Kanezaki [29]. The method iteratively updates pixel labels and network parameters. The initial labels are generated through clustering and refined by merging similar superpixels. The refined pixel labels are used to calculate the loss between the network responses and the refined cluster labels. Error signals are then backpropagated to update the parameters of convolutional filters and the classifier. [29]. One of the limitations of this approach is that boundaries are fixed in the superpixel extraction process [34].

To mitigate this limitation, a spatial continuity loss is proposed by Kim et al. [34]. The method, based on differentiable feature clustering, utilizes a convolutional neural network (CNN) to extract deep features from input images, which are then clustered in a q-dimensional cluster space using a one-dimensional (1D) convolutional layer. The response vectors of the features in the cluster space are normalized and used to obtain the label map of the input image. The label map is refined iteratively by minimizing the combination of similarity loss and spatial continuity loss. Overall, the proposed method offers improved segmentation accuracy, faster convergence, and a simpler initialization process compared to previous works [34].

Many other approaches in deep unsupervised learning leverage clustering techniques [51, 26, 24]. Mutual information maximization, which involves enhancing the information shared between variables, has also been frequently used to improve feature learning and representation quality [47, 22, 44].

4. Void-Finder Algorithm

As shown in Chapter 2, dust voids are particularly challenging to identify since the complex distribution of the ISM requires the dust density changes to be evaluated locally, and because the potential voids are not fully encapsulated with a complete dense layer of dust in all directions. Therefore, a successful method for identifying these voids must not only assess local dust variations but also possess a broader perspective on the overall structure of the void surroundings to accurately delineate its boundary.

A common element that can be seen in both the *Per-Tau Shell* and the *Cepheus Flare shell* is the pattern of their mean radial density profile. The mean radial density profile describes how the density changes as a function of distance from a central point. The density is averaged over an area or volume, providing an overview of the distribution rather than focusing on specific points. This averaging helps smooth out fluctuations or irregularities in the density distribution. For both voids, we see that the density is smallest near the center, peaks at the supposed void borders and diminishes once again past the void's boundary. This process is indicative of supernova remnants where the central explosion pushes gas and dust radially outwards, aggregating dust density as one moves away from the supernova site. At some point, where the supernova influence has yet to reach the surrounding ISM, the density gradually drops to the average surrounding ISM density. Given initial density variations in the ISM, parts of the supernova outflow can escape through low density regions, generating holes in the dust envelope.

The Void-Finder algorithm leverages this specific characteristic to identify potential supernova void candidates. Its core concept involves assuming any voxel in a 3D dust map as the center of a void, computing the mean radial profile outward, and assessing if the resulting distribution meets the predefined criteria for void classification. This approach effectively tackles the two key challenges faced by traditional image segmentation algorithms: identifying local regions with lower density at the center compared to the surroundings, and delineating the void area despite the presence of low-density regions within the surrounding dust.

Though the idea is quite simple, the algorithm consists of multiple stages to ensure the quality and robustness of the selected candidates while minimizing the processing time. The first decision point arises when selecting the voxels that will serve as centers for the mean radial profile evaluation. Though the most thorough option is to check every single voxel in the 3D dust map, that requires a lot of computations and is often redundant, as neighboring voxels will likely result in similar radial profiles and, thus, void candidates. This is especially true when the void size is much larger than the distance between voxels.

An alternative approach involves selecting evenly spaced points throughout the grid. In this scenario, the ideal distance between points would depend on factors such as the dust map resolution and the minimum void candidate radius desired. This strategy can

significantly reduce processing time while remaining quite effective. Even in cases where the selected seeds slightly deviate from the true center of the void, the radial mean profiles remain robust to small changes in the center location. A visual representation of this concept is provided in Figure 4.1, illustrating a 2D example where input seeds are spaced five pixels apart from each other, for a void with an approximate radius of 80 pixels. Upon the algorithm's execution, 43 seeds are identified as potential void centers. However, it becomes apparent that all these points correspond to the same void. This underscores the necessity for a merging step to ensure that only distinct candidates are further evaluated.

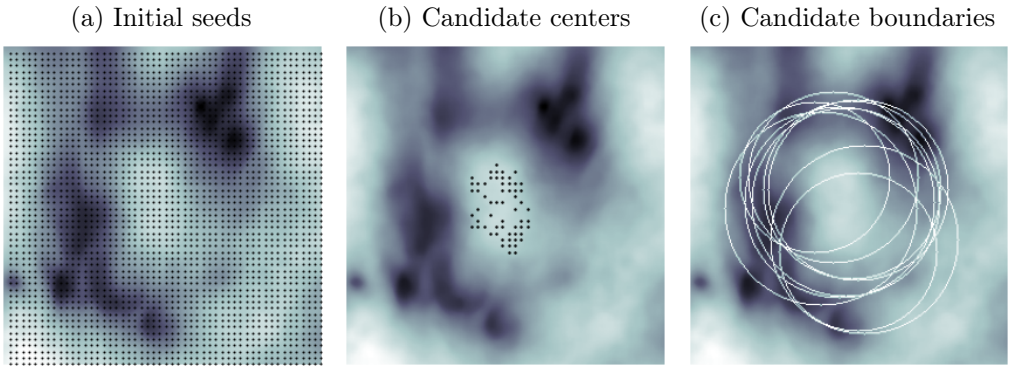


Figure 4.1.: Void-Finder candidate selection. Image (a) illustrates the initial seed grid for a specified region in the dust map containing a void. Image (b) showcases seeds that have been identified by the algorithm as potential void candidates. Image (c) depicts the approximation of some candidate boundaries.

After evaluating the seeds, the merging step removes duplicated candidates by assessing the overlapping volume of the approximated spheres (or circles in the 2D example) using a predefined threshold. Candidates that exhibit an overlapping volume above this threshold are considered duplicates and subsequently eliminated from further consideration.

After removing duplicated candidates, the next step focuses on enhancing the characterization of the voids represented by the remaining seeds. While spheres serve as a valuable approximation for filtering overlapping candidates, they fall short in accurately representing the true shape of voids. For this purpose, ellipsoids are a better choice, as they can accommodate variations in radius across different directions caused by various physical processes such as Galactic shear and anisotropic gas expulsion. In this phase, the density of the local area containing each candidate is evaluated, and an ellipsoid is fit to points that exhibit a density higher than the local mean. In this context, the surface of the ellipsoid represents the dust envelope that surrounds the void. While the ellipsoid delineates the boundaries of the void, a set of measures considering density ratios and distribution allows for further selection of the candidates' desired characteristics.

To fit the ellipsoid, multiple points are sampled from the sphere's surface, and their distances to the center are updated to place them at the densest region along that line.

This process is illustrated in Figure 4.2. Since the ellipsoids' extent might be larger towards a given direction compared to the initial sphere, its position may change and it becomes possible to have more than one candidate representing the same void again. Hence, a second merging step is applied, this time based on the ellipsoid's proportions and proximity to other candidates. The metrics derived from the ellipsoids are also taken into account, and are used for selecting the ellipsoid that best represents the void candidate. An overview of the algorithm stages is shown in Figure 4.3.

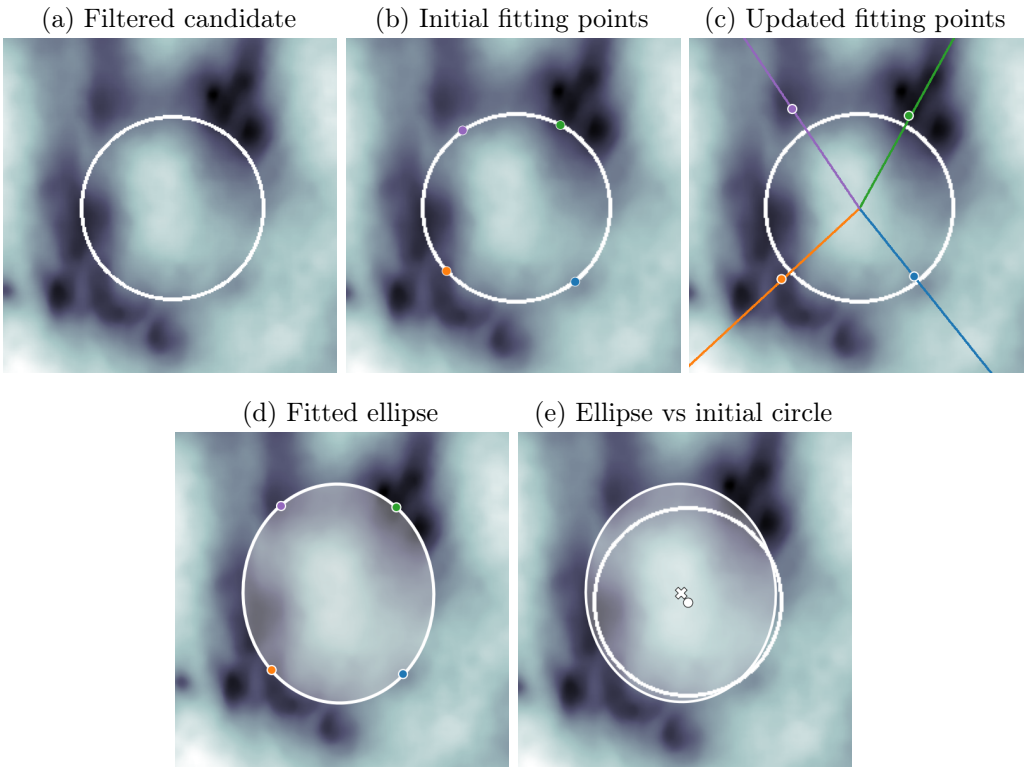


Figure 4.2.: Void-Finder ellipsoid fitting process. Image (a) shows a selected void candidate after the first merging step. Image (b) exhibits the points selected for the ellipse fitting. Image (c) depicts how the distances of these points from the center are updated so that they lie in the densest nearby region. Image (d) shows the ellipse that was fitted based on the selected points. Finally, image (e) displays the initial circle and its center (round point), as well as the fitted ellipse and its center (cross).

The upcoming sections will provide a more comprehensive description of each algorithm step, offering further details on its execution.

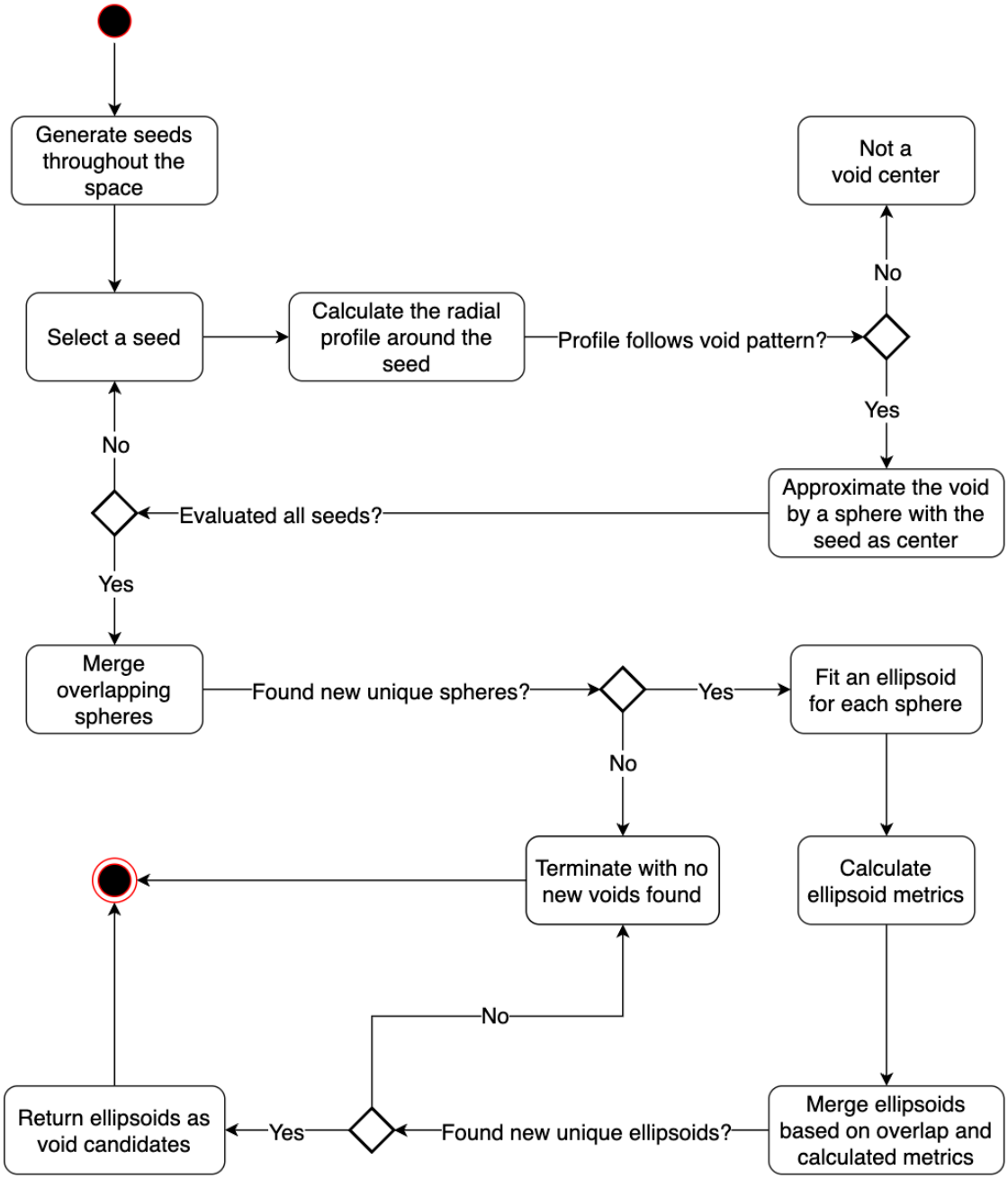


Figure 4.3.: Activity diagram of the Void-Finder algorithm.

4.1. Seed selection

The seed selection serves as the initial step of the algorithm. As previously mentioned, it's not essential for the selected seed to precisely correspond to the true center of a void candidate, as the center location can shift during the process. However, it is essential that there are sufficient seeds distributed throughout the data space so that seeds close to void centers are identified by the algorithm. Ideally, the seeds would also be spaced out enough to minimize the occurrence of duplicates. Although the optimal number of seeds varies for each dataset, one of the main factors to be taken into account is the expected size of void candidates. When searching for large voids, fewer seeds can be created because it is expected that the centers are farther apart from each other.

An alternative to employing a regular grid with a fixed distance between seed points is to utilize a Sobol sequence for point selection, as proposed by Sobol [55]. The Sobol sequence generates points in a space with low discrepancy, resulting in a more even distribution compared to random sampling methods [55]. The main advantage of using this method is the possibility to initiate the algorithm with a set of points and subsequently incorporate additional seeds for incremental runs, while considering the locations of previously selected seeds to avoid overlap. This process can be repeated until no new void candidates are found. The incremental seed selection is visualized in Figure 4.4.

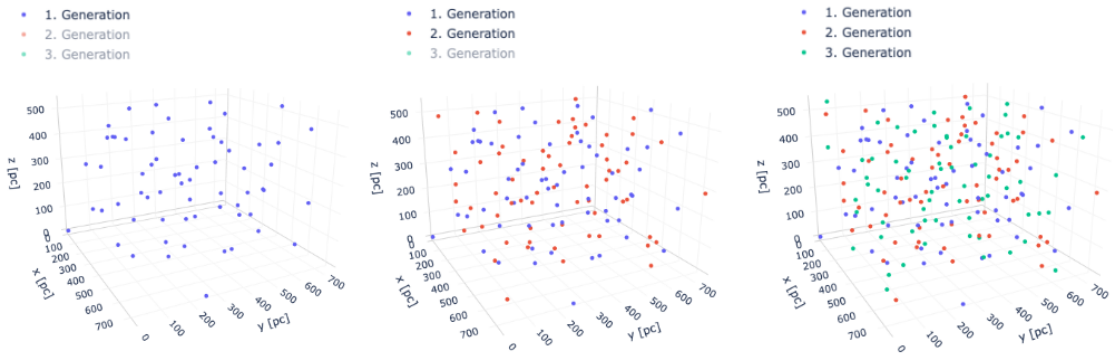


Figure 4.4.: Visualization of seed generations with the Sobel sequence. On each step 6^2 new seeds are generated, with high sparsity and minimal overlapping.

4.2. Selection criteria

Each of the selected seeds are considered a center point for the mean radial profile calculation. The mean radial profile measures the average intensity of voxels that are at the same distance from the center, grouping them into bins that form concentric rings around the center point. The average intensity of each ring is stored in an output array, where each position d corresponds to the ring located at the distance d from the center. This is illustrated in radial mean profiles of the Per-Tau Shell 2.7 and the Cepheus Flare Shell 2.9.

A key aspect of the algorithm lies in the distinctive characteristics of the radial profile of an ideal void, which serve as guidelines for identifying other void candidates. As explained in Chapter 2, the physical processes of a supernova explosion shape the ISM in a characteristic way, where the mean radial profile of existing voids exhibits the following the pattern:

a) low density near the center, gradually increasing as the distance from the center increases,

b) a density peak that marks the boundary of the void, and

c) a decrease in density beyond the peak, where the ISM was unaffected by the void.

Since fewer voxels are considered near the center of the mean radial profile, the beginning of the profile curve may exhibit noise. To mitigate these noise effects, the radial profile function is smoothed before the gradient can be used to determine whether the void candidate fulfills all the criteria of the previously presented pattern. The comparison starts by counting the number of consecutive points in the gradient that are positive, which characterizes the low-density center with increasing density outward (a). When the derivative is closest to zero, the radius of the candidate is estimated, as this corresponds to the peak density in the region (b). Negative gradient points correspond to a subsequent decrease in density (c). An example for the smoothed gradient compared to the mean radial profile of the Per-Tau shell can be seen in Figure 4.5.

Additional specifications such as the minimum desired void radius and the minimum recognized rate of change from center to peak also play a role in the candidate selection process. In total, six parameters can be defined for a void candidate evaluation:

- *Minimum descent:* the minimum number of points that are expected to decrease after the peak is defined, as a percentage of the total number of points that were ascending before the peak.
- *Minimum radius:* the minimum radius accepted for the void candidate. The algorithm will discard any voids that are too small according to this parameter.
- *Maximum radius:* the maximum radius used in the search for the void candidate. This is the radius used in the calculation of the radial profile of each seed.
- *Minimum density change:* (optional) indicates the minimum accepted rate of change in density from the start to peak of the profile. Higher values result in more pronounced void candidates.

When the radial profile of a seed point meets all the criteria, the seed is designated as the center of a void candidate. This void extends up to the radius r corresponding to the peak of the profile, approximating the void candidate as a sphere. For the cases when the seed point is not located inside a void, these criteria are generally not met, and the point is discarded from further consideration.

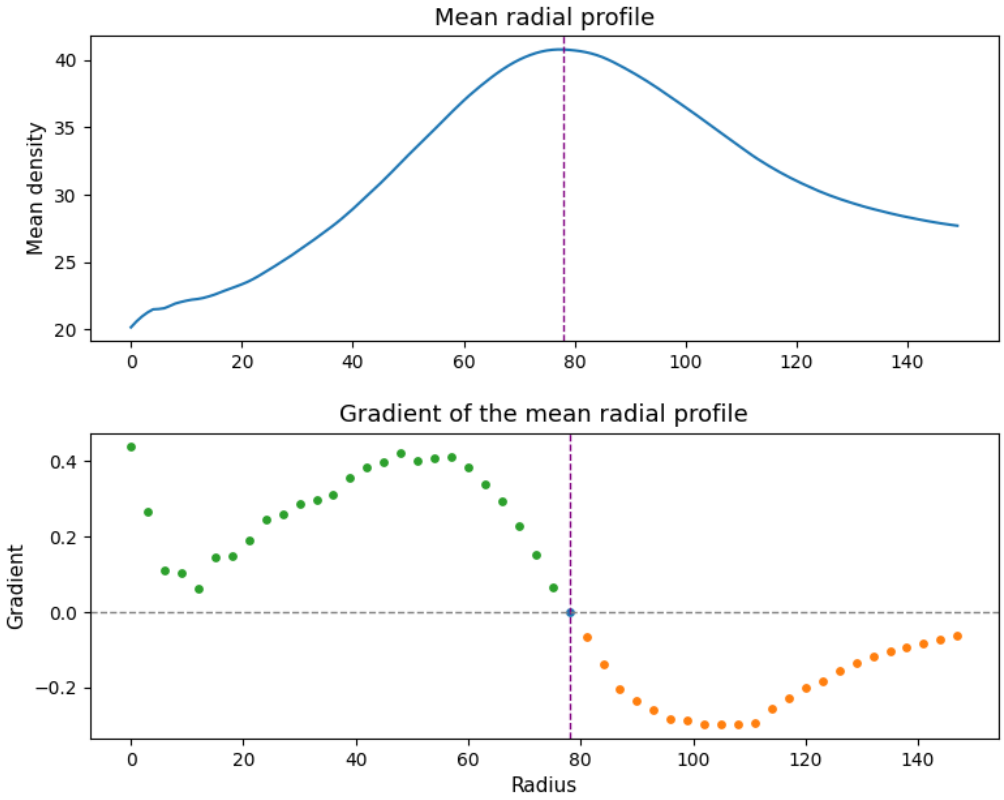


Figure 4.5.: Mean radial profile and the smoothed gradient for the Per-Tau shell. The vertical line marks the density peak. Gradient points above zero (green) correspond to a density increase in the profile, while points below zero (orange) represent a decrease.

4.3. Candidate merging

After the seeds that fulfill the criteria are identified, it can be that multiple seeds represent the same void. That can occur when the distance between the seeds is too small in relation to the total size of a void. In order to avoid repeated calculations for such cases, the extra seeds that likely represent the same void are removed and only one remains. That is done based on the previously mentioned radius, assuming a spherical shape. If the volume overlap of two candidate spheres exceeds the threshold t, they are considered to represent the same void.

The overlap between the two approximated spheres is calculated with the Jaccard index [25], also known as the Jaccard similarity coefficient. It is defined as the size of the intersection divided by the size of the union of the sample sets. In the context of

calculating the overlapping volume between two spheres, the Jaccard Index quantifies how much of the two volumes overlap relative to their combined volume. Mathematically, if A and B represent the volumes of two spheres, the Jaccard Index $J(A, B)$ is given by the formula

$$\left(J(A,B) = \frac{|A \cap B|}{|A \cup B|} \right),$$

where $|A \cap B|$ denotes the volume of the intersection of the spheres and $|A \cup B|$ denotes the volume of their union [25]. The threshold value lies in the range $0 < t <= 1$, with 1 indicating that both spheres overlap perfectly and 0 indicating that there is no overlap between the spheres.

4.4. Ellipsoid fitting and density metrics

After the candidate merging step, the algorithm is left with a list of void centers and their potential sizes. Given that real voids may have ellipsoidal shapes, assuming them to be spheres is an imperfect approximation. To refine the estimation and better align with the expected characteristics of a void, an ellipsoid shape is fitted to the data. While still an imperfect shape, the ellipsoid significantly improves the estimation by allowing for different radii in each dimension of space.

The ellipsoid aims to identify the densest points along the void's perimeter. To locate the density peaks in any given direction, a line is cast outward from the center of the void candidate. The density values of the voxels along this line follow a pattern similar to the radial profile curve but without the averaging effect, as shown in Figure 4.6. This process enables the identification of density peaks in multiple directions, effectively outlining the void's perimeter. The cast line extends from the sphere's center to a distance of twice its radius, allowing the ellipsoid boundaries to expand beyond the initial sphere. The ellipsoid is then fitted to the peak point of each line. While most lines lead to the void's boundary, some may not due to missing sections of the boundary. To exclude such outliers, only points within two standard deviations of the overall distribution are included in the fitting process. This process corresponds to the steps shown in Figure 4.2 for the 2D case.

With the fitted ellipsoid, it is possible to derive several metrics for further void characterization, including:

- Shell density: indicates the dust coverage around the void. It is calculated as the ratio of high-density voxels on the ellipsoid surface to its total surface area. A voxel is considered high-density if its density is above the mean density of the evaluated region (2 times the initial sphere radius). The shell density ranges from 0 to 1, with 1 representing a void completely surrounded by dust in all directions. Shown in Figure 4.7a.

- Inner density: represents the ratio of high-density voxels inside the ellipsoid compared

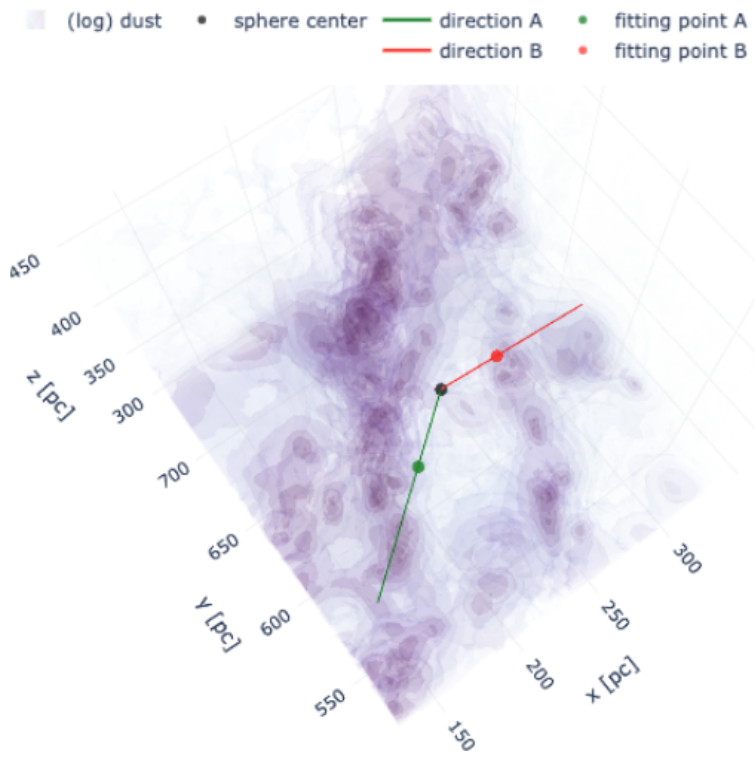
to its total volume. It ranges from 0 to 1, with 0 indicating a void where all voxels have densities below the mean density of the evaluated region.

- Density ratio: this metric is calculated as the ratio of shell density to inner density. It indicates the proportion of dust surrounding the void relative to the dust within the void.
- Closedness: this metric measures how uniformly the dust is distributed around the void. It is calculated as the distance from the ellipsoid center to the center of mass of the high-density voxels on the ellipsoid surface. If there is a higher concentration of dust on one side of the void, the center of mass shifts towards that side, increasing the distance from the center of the ellipsoid. This is shown in Figure 4.7b.

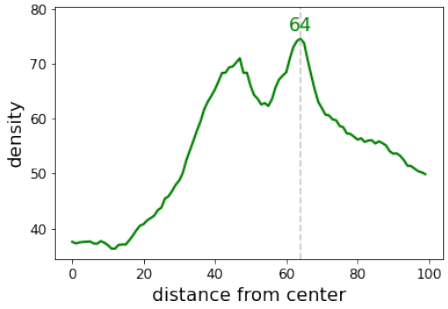
During the ellipsoid fitting process, it is likely that the center location of the void candidate changes in comparison to the initially estimated sphere. This results in a higher chance that more than one ellipsoid represents the same void. To output unique void candidates, a second merging step is applied, this time based on the ellipsoids. The distances of the ellipsoids centers are compared and they are considered to overlap if the center of one ellipsoid lies inside another ellipsoid. When this happens, the ellipsoid with highest shell density is selected. Alternatively, other metrics can be used for the final ellipsoid selection. While not all of the metrics are necessarily used by the algorithm for selection purposes, they are able to provide extra information that domain experts can use to filter void candidates that best meet their specific criteria. An example can be seen in Figure 4.7b, where the center of mass can be used for filtering out open voids.

Having outlined the proposed method and its capabilities, the following chapter applies these techniques to real data, providing a comparative analysis to evaluate the practical effectiveness in identifying dust voids.

(a) Dust map



(b) Densities along line A



(c) Densities along line B

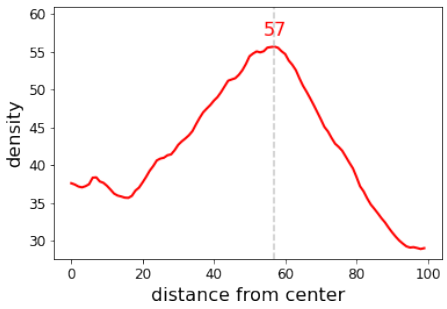
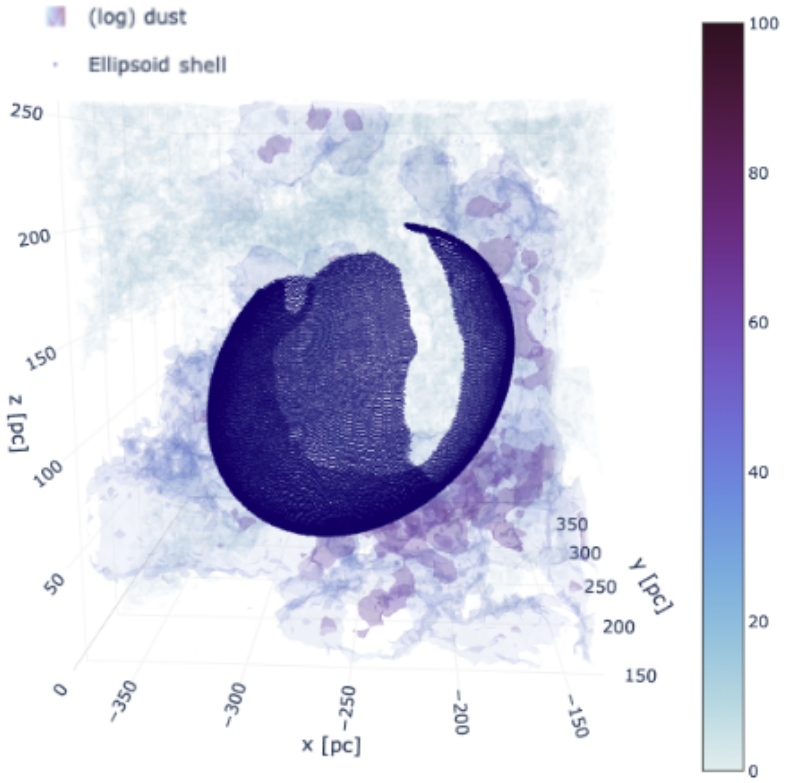


Figure 4.6.: Density peaks in random directions. Image (a) displays the dust map focused on the Cepheus Flare region, including the estimated center and two sample lines used for the ellipsoid fitting step. Lines are drawn in arbitrary directions and the location where the density peaks along the line is used as a fitting point for the ellipsoid. Images (b) and (c) show the density along the lines, highlighting the density peaks.

(a) Shell density for a fitted ellipsoid



(b) Center of mass of dust shell vs ellipsoid center

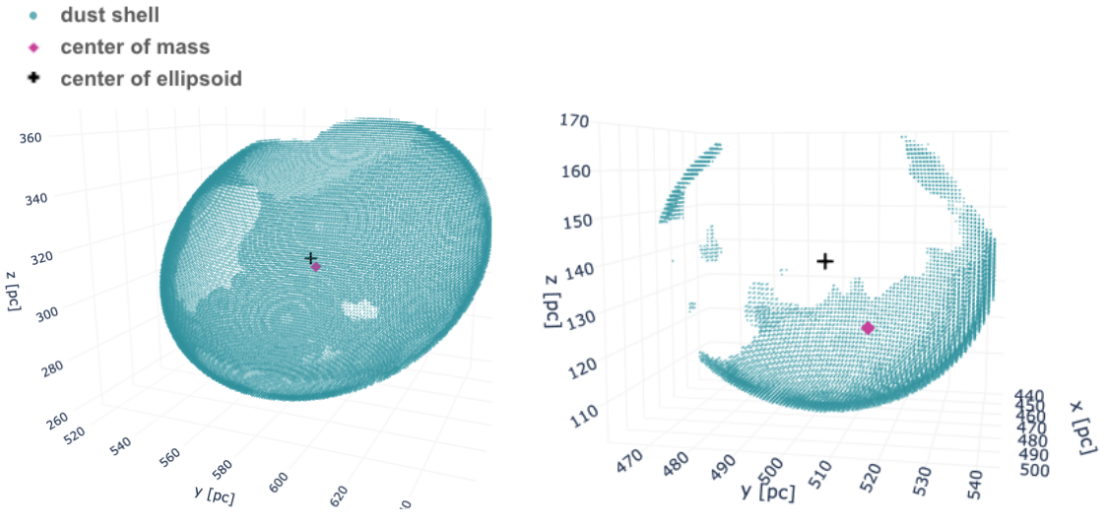


Figure 4.7.: Ellipsoid density metrics. Image (a) shows how the shell density correlates to the dust in the region, making open areas easier to spot. Image (b) illustrates how the distance between the center of mass of the shell and the ellipsoid center changes when the shell density is balanced (left) or unbalanced (right).

5. Results

In this chapter, the image segmentation methods discussed in Chapter 3, along with the Void-Finder algorithm introduced in Chapter 4, are applied to the simulation data and to the dust map. By systematically applying these methods on the datasets, the performance and effectiveness of the Void Finder algorithm in detecting voids is compared to the other image segmentation methods, providing insights into the strengths and weaknesses of each approach in identifying voids in the three-dimensional dust map.

Although the methods discussed in Chapter 3 were thoroughly considered for void identification, not all of them are included in this analysis. The selection of methods was influenced by their availability in public Python libraries, ensuring ease of implementation and reliability. The following section explains the rationale behind the selection process and highlights the specific algorithms chosen for the comparison based on their accessibility and relevance to the task.

5.1. Selected methods

Among the thresholding approaches, Otsu's method was chosen as a global technique for its well-established nature, simplicity and automatic threshold selection. Its robustness in separating foreground from background can align with the task of identifying voids in three-dimensional dust maps. Furthermore, its widespread usage and availability in popular libraries such as *scikit-image* make it a convenient choice for processing large datasets.

Sauvola's method was chosen as a local thresholding approach for its dynamic range capabilities, which can be an advantageous choice for segmenting regions with varying intensity levels. Local methods are likely to yield better segmentation results overall, as low and high dust densities are relative terms defined by local context in the dust map. This combination of global and local approaches establishes a baseline for the comparison of more advanced image segmentation algorithms.

Edge-based methods were not included in the current evaluation due to the specific characteristics of the data. The 3D dust maps exhibit many local fluctuations, and the definitions of high and low dust densities are inherently local, making edge detection less effective. Furthermore, the presence of open voids that are not completely surrounded by dust complicates the identification of clear boundaries that edge-based methods rely on. These methods typically perform best in scenarios where distinct, continuous edges separate different regions, which is not the case of the interstellar medium represented in the dust map.

As a region-based approach, the *ConfidenceConnectedImageFilter* implementation from

the Insight Toolkit (ITK) [70] was chosen for this evaluation. This method is particularly interesting because, unlike other region-growing methods available in the package, it does not require any manual threshold values to be set, which simplifies the segmentation process. The algorithm begins with the selection of one or more seed points within the image. Then, the mean and standard deviation of the voxel intensities within a small neighborhood around these seeds are calculated, and these statistics form the basis for determining whether neighboring voxels should be included in the region. The neighboring voxels' statistics are calculated and adjusted by the user-defined multiplier parameter, which controls the width of the confidence interval. If the voxel's intensity falls within this confidence interval, the voxel is added to the region and the mean and standard deviation of the region are updated. This process is repeated iteratively, stopping when no more voxels are added to the region, or when a maximum number of iterations is reached [27].

In the deformable models category, the chosen method is the morphological Chan-Vese implementation from the *scikit-image* package. The method uses morphological snakes, which behave similarly to active contours but differ by applying morphological operations, such as dilation and erosion, on a binary array. This contrasts with traditional active contours, which solve partial differential equations (PDEs) on a floating-point array [39]. This results in faster and more numerically stable segmentation, which can also be used in 3D data. Additionally, the morphological variation of the Chan-Vese method is still effective for objects with ill-defined edges, relying instead on different voxel value averages inside and outside the segmented object [12].

For the clustering, K-means implementation from *scikit-image* was used. Initially, the density data, which follows a single Gaussian distribution, proved insufficient for meaningful segmentation. To enhance the clustering process, additional features inspired by multidimensional transfer functions [35, 4] used in volume rendering were explored. Some of the features tested include first and second derivatives, which are generally good for identifying borders, and entropy, for texture. These additional features provided a more nuanced understanding of the data structure, leading to improved segmentation results. This approach allowed for a more effective differentiation of regions within the 3D dust map and simulation.

Graph-based methods were not included in the current evaluation due to the lack of readily available algorithms that are efficient enough to handle the datasets. While these methods can be powerful for segmenting complex structures by leveraging relationships between data points, their computational demands are significant, especially for large-scale 3D data. Consequently, these methods were deemed impractical for the current purposes.

As an unsupervised deep learning approach, the available implementation of the differentiable feature clustering algorithm [34] provided by the authors was used. Since the implementation only accepted 2D images, an adaptation for the usage with 3D images was performed, which mainly updated the data structure elements and did not affect the overall process of the method.

The next section details how these methods were applied to the available datasets.

5.2. Evaluation setup and parameter spaces

To ensure that each method is evaluated under the most favorable conditions, parameters were systematically explored and optimized to maximize the Jaccard score between the output segmentation and the labeled ground truth of the simulation. This was done to estimate the best-case scenario of the compared image segmentation methods and to avoid poor performance caused by suboptimal parameter choices.

For the dust map, where ground truth data is unavailable, the validation criteria focus on the effectiveness of the algorithms in segmenting the Per Tau shell and the Cepheus-Flare shell, the two known voids discussed in Chapter 2. Instead of manually testing various parameter sets, known information about these voids, such as their estimated locations and radii, was used to create approximate spherical models of the structures. These spherical models, although not as precise as a ground truth, allow the automated comparison to fine-tune the parameters, following a process similar to that used with simulations and ground truth.

Table 5.1.: Parameter settings

Method	Parameter	Search range	Best value simulation	Best value dust map
Sauvola	window size	1-145	123	137
	k	0-1	0.15	0.25
	r	0-1	0.93	0.94
Region growing	initial neighborhood radius	1-100	11	90
	multiplier	1-5	2.85	2.05
	iterations	1-15	1	11
Morphological Chan Vese	smoothing	1-4	1	2
	lambda1	0-1	1	0.1
	lambda2	0-1	1	0.65
	iterations	1-50	44	4
Deep learning (manual search)	stride	1-2	1	1
	learning rate	0.001-1	0.006	0.003
	step size for similarity loss	0-1	0.7	0.6
	step size for continuity loss	0-1	0.3	0.4
	number of channels	1-100	2	2
	number of convolutional layers	1-5	3	5
	iterations	500-1000	730	600
Void-Finder	window size	4-10	6	6
	step	1-3	2	2
	minimum descent	0-0.5	0.4	0.3
	minimum radius	10-30	25	20
	minimum density change	0-0.8	0.3	0.25
	merge overlap threshold	0.3-1	0.5	0.4

To optimize the parameters of the selected methods for the simulation data, the *gp_minimize* [23] function from the scikit-optimize (skopt) library was employed. This function utilizes Gaussian process-based optimization, a technique for finding the minimum of an objective function, to minimize the complement of the Jaccard similarity between the segmented output and the ground truth labels. This approach ensures that the segmentation methods are tuned to deliver the best possible performance relative to the ground truth, as the objective is to gauge the effectiveness of each method in finding voids in the dataset. The parameters of each method are described below.

The search range and optimal parameters found are summarized in Table 5.1.

Global threshold (Otsu)

As an automatic method, no parameters were optimized.

Local threshold (Sauvola)

- **window size**: defines the local area for the calculations.
- **k**: controls the effect of the standard deviation.
- **r**: normalization parameter for the standard deviation.

Region growing (ConfidenceConnectedImageFilter)

- **seeds**: defines the local area for the calculations. For the evaluation of the simulations, the seeds were set to be the known supernovae locations, while for the dust map it was set in the known voids.
- **initial neighborhood radius**: the initial radius of the neighborhood within which the statistics are calculated.
- **multiplier**: controls the confidence interval (confidence interval = mean $\pm$ multiplier $\times$ standard deviation). Can be any value bigger than zero, with a typical value being 2.5.
- **iterations**: number of times the process repeats.

Deformable models (Morphological Chan Vese)

- **initial level set**: the initial level set from which the level set will evolve. For this evaluation, the resulting segmentation image from Otsu's method was used.
- **smoothing**: number of times a smoothing operator is applied. According to the scikit-learn documentation, values between 1-4 are reasonable.
- **lambda1**: outer region weight parameter. Relative to lambda2, defines whether outer region or inner region have a larger range of values.

- **lambda2**: inner region weight parameter.
- **iterations**: number of times the process repeats.

  Due to the relatively long processing time for the full dataset, the parameter search and final segmentation were conducted on a downscaled input. This reduced the data by a factor of 8, from the original dimensions of (512,512,512) to (256,256,256) for the simulations and from (740,740,740) to (370,370,370) for the dust map.

Clustering (K-Means)

The clustering was evaluated with different combinations of features that were used to represent the image. These include:

- **spacial coordinates**: the x, y, z coordinates of the space.
- **intensity values**: representation of the dust density.
- **entropy**: indicates how surface variations are distributed, with higher entropy corresponding to more complex or rough textures, while lower entropy indicates smoother, more uniform areas.
- **first derivatives**: represents the gradient of the image, indicating the presence and direction of edges.
- **second derivatives**: measures the rate of change of the gradient, helping refine edge detection and identify small, detailed features in an image.
- **distance from closest peak**: measures the distance of each voxel to the closest peak, which was defined here as any value higher than $mean + 2 * std$.

For both the simulations and the dust map data, the features that yielded the best segmentation included the spatial coordinates, the intensity values, and the entropy, weighted at a ratio of $[1, 3, 2]$, respectively. To simplify the process, the number of clusters n were set to $n = 2$ so that the resulting segmentation would be binary and therefore easier to evaluate against the ground truth.

Deep learning (differentiable feature clustering)

The differentiable feature clustering algorithm has shown a high computational demand associated with processing the 3D data, in particular for the memory requirement. This has made it impractical to perform a parameter search as done with the other methods. To mitigate this issue, the method was applied to a downscaled version of the data (factor of 27), which significantly reduced the computational load and made the process feasible. Subsequently, the parameter space was explored manually to identify good settings for the algorithm. This approach ensured that the algorithm could still be evaluated within the constraints of the available computational resources, while also providing insights into its performance in the context of void identification in 3D dust maps. The tested ranges and the selected values can also be seen in Table 5.1

5.3. Simulation results

The simulation dataset contains 138 simulated supernovae explosions in various locations, time frames and intensities. Oftentimes multiple supernovae occurred in the same area, further expanding the boundaries of voids left by its predecessors. With the labeling of the simulation data as a ground truth, a total of 24 voids could be completely separated from the background, covering the location of 81 supernovae. Because of the different time frames of the explosions and their different intensities, the voids left by some early explosions with low intensities faded away as time passed and other stronger supernovae occurred that altered the dust distribution nearby. From the 24 voids labeled, 11 are located where supernovae occurred, while the remaining ones are areas that have the characteristics of a void but are not related to supernovae explosions.

With the current ground truth there are two possible ways to evaluate the results. From a pure image segmentation view, a similarity measure, such as the Jaccard score, can be used to compare the overlap of the segmentation from the methods with the ground truth. From a domain perspective, however, it is more relevant to compare how many different objects can be obtained from the segmentation, and from those evaluate how many can be attributed to supernovae events.

In regards to the Jaccard score, which quantifies how closely the segmentation matches the ground truth, the deep learning algorithm performed best, followed by the clustering method. Among the traditional image segmentation methods, region growing achieved the lowest score. The Void-Finder method has the lowest Jaccard score overall, which is expected since its primary goal is not to precisely delineate the void's detailed shape but to indicate its location and size with an approximated ellipsoid. The segmentation for the simulation are shown in Figure 5.1, while the scores for each method are summarized in Figure 5.2.

For the domain perspective, Figure 5.3 shows information about the supernovae found in the segmented regions of each method on the simulations dataset. Each resulting segmentation was labeled to identify distinct objects that represent the voids. Within the identified voids, some are related to supernovae events, while others have the characteristics of a void without having nearby supernovae explosions. In the figure, the blue region shows the voids that are related to supernovae, while the orange region indicates the voids that are not related to supernovae events. The sum of these elements in each bar corresponds to the total voids found by the method. For this analysis, only objects with volume bigger or equal that of a sphere of radius 10 were kept, as a filter to exclude small voids and noise. From this perspective, the Void-Finder algorithm excels at detecting objects associated with supernovae, which are the voids astrophysicists aim to study. The deep learning algorithm also performed well, identifying the same number of objects with supernovae as the ground truth.

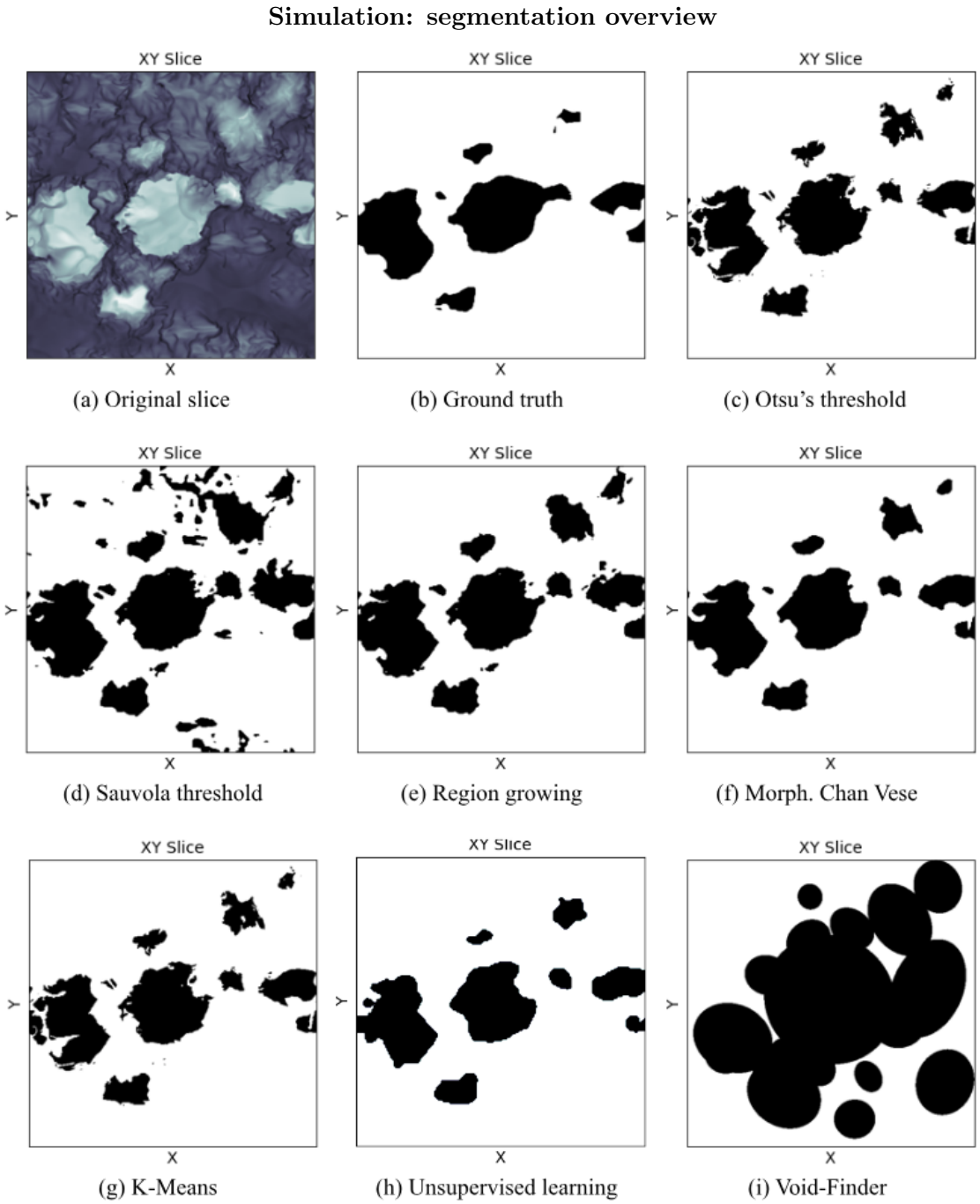


Figure 5.1.: Slices of the simulation data and segmentation results for the different applied methods.

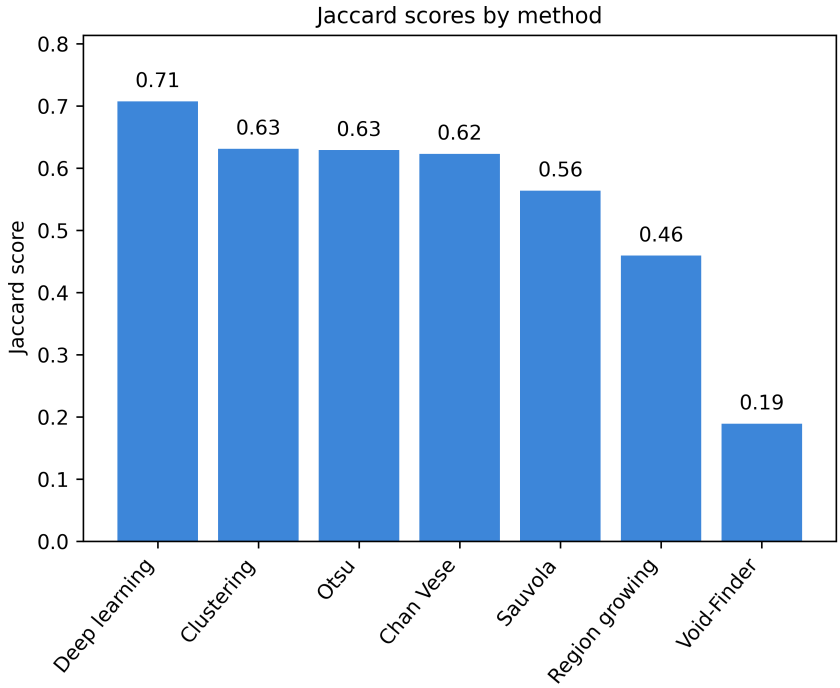


Figure 5.2.: Jaccard scores for simulations dataset.

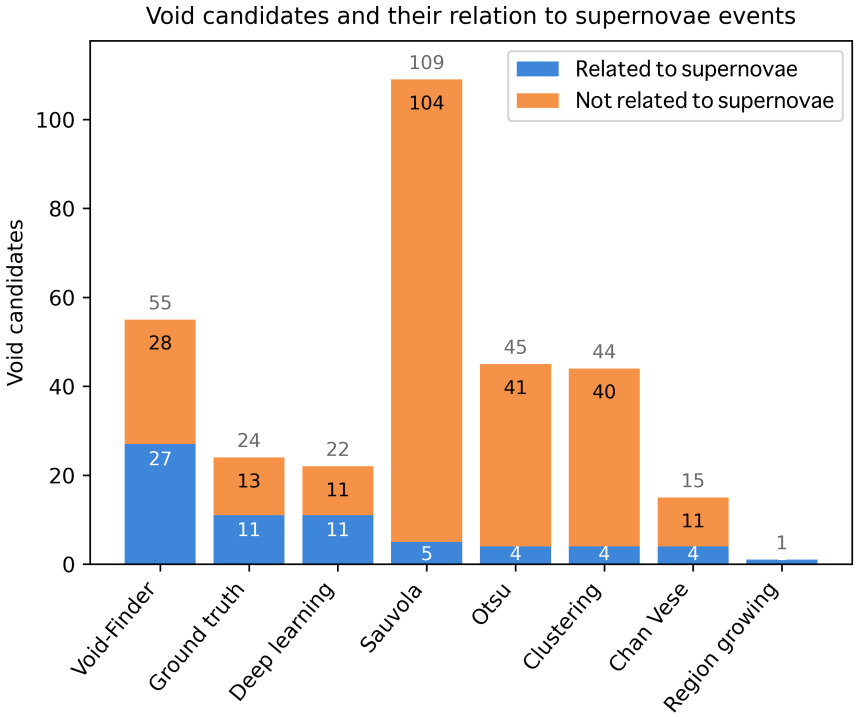


Figure 5.3.: Overview of void candidates and their relationship to known supernovae events in the simulation dataset.

The fact that the Void-Finder has found more distinct objects containing supernovae than the labeled ground truth, shows one of its main advantages against image segmentation methods. While the segmentation methods fail to fully separate voids when their borders are close together, the Void-Finder can still separate them. This can be seen in two of the biggest bubbles in the simulation in Figure 5.4. Image (a) shows the middle slices of the ground truth, where it is apparent the presence of two voids. Image (b) shows the projection of that object through all the slices, and make it possible to see that the two objects have a small connection, as well as a connection to a third void at the right corner that could not be seen in the slice in (a). This has led the manual labeling tool and other methods to recognize this as a single void. However, the Void-Finder has identified two distinct objects, as shown in image (c).

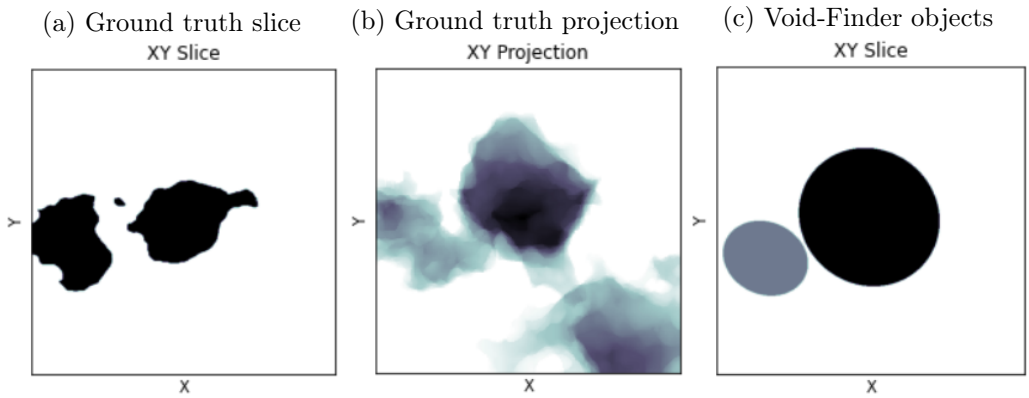


Figure 5.4.: Void delimitation of ground truth and Void-Finder. Image (a) shows a slice of the ground truth, where two voids can be seen. Image (b) shows the projection of that object through all the slices. Image (c) shows the slice with the corresponding objects found by the Void-Finder algorithm.

5.4. Dust map results

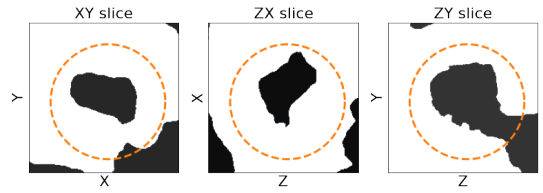
On the dust map, the main validation criteria is the identification of the Per Tau shell and the Cepheus-Flare shell as void candidates. To perform this evaluation, the resulting segmentation of each method was labeled to identify distinct objects, each representing a potential void. To remove noise and small objects, only labels with a volume larger than that of a sphere with a radius of 5 are kept. These objects are referred here as "significant objects". The slices and projections of each method for the regions of the Per Tau shell and Cepheus-Flare shell are presented in Table 5.2. An overview of the significant objects identified is shown in Figure 5.5.

Dust map validation

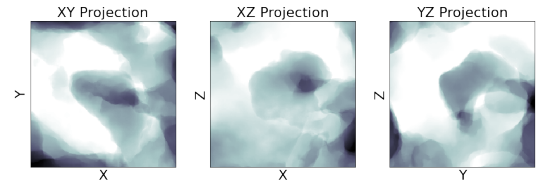
Otsu's threshold

Per-Tau Shell

(a)

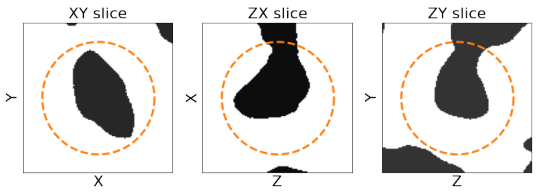


(b)

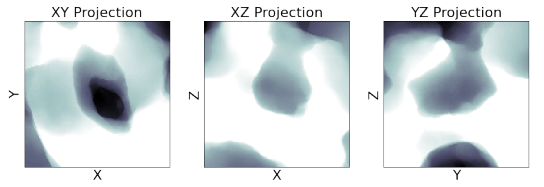


Cepheus Flare Shell

(b)



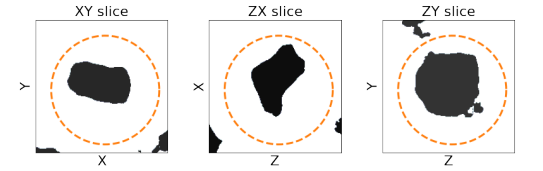
(d)



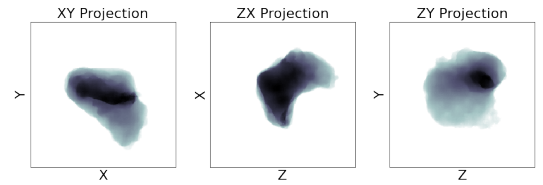
Sauvola threshold

Per-Tau Shell

(a)

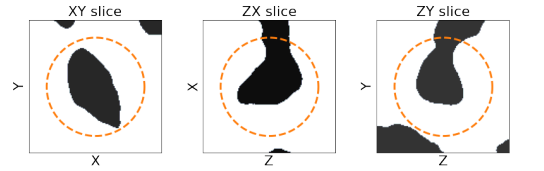


(b)

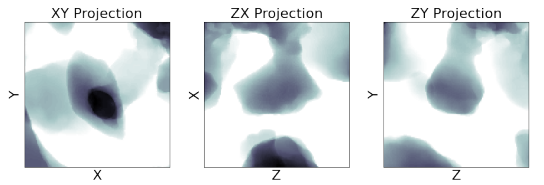


Cepheus Flare Shell

(b)



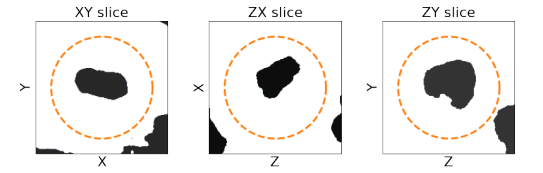
(d)



Confidence connected region growing

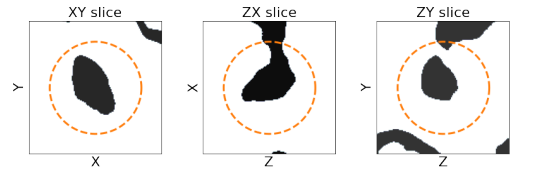
Per-Tau Shell

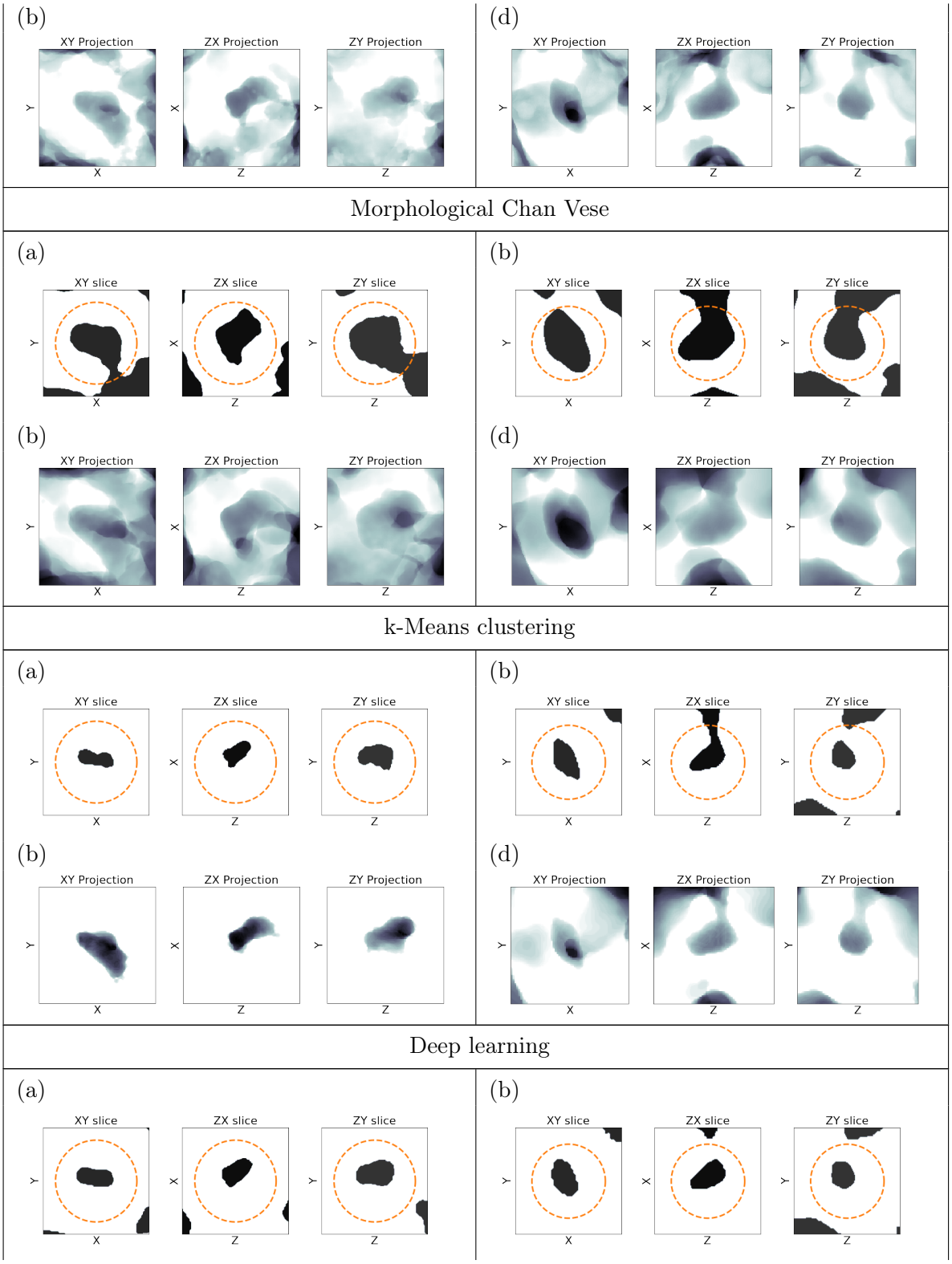
(a)



Cepheus Flare Shell

(b)




(b)
XY Projection
ZX Projection
ZY Projection
(d)
XY Projection
ZX Projection
ZY Projection
Morphological Chan Vese
(a)
XY slice
ZX slice
ZY slice
(b)
XY Projection
ZX Projection
ZY Projection
(b)
XY slice
ZX slice
ZY slice
(d)
XY Projection
ZX Projection
ZY Projection
k-Means clustering
(a)
XY slice
ZX slice
ZY slice
(b)
XY Projection
ZX Projection
ZY Projection
(b)
XY slice
ZX slice
ZY slice
(d)
XY Projection
ZX Projection
ZY Projection
Deep learning
(a)
XY slice
ZX slice
ZY slice
(b)
XY slice
ZX slice
ZY slice

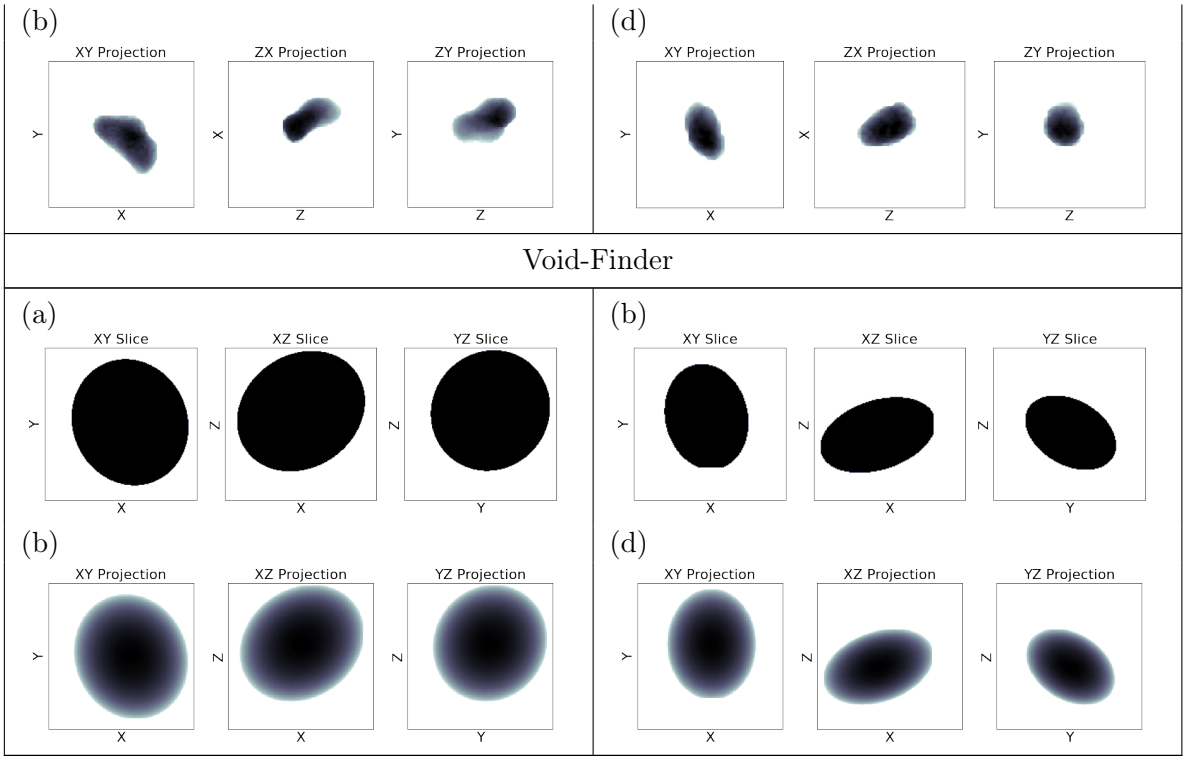


Table 5.2.: Resulting segmentation of the applied methods for the Per-Tau shell (left) and Cepheus Flare shell (right). Images (a) and (b) display the slices of the resulting segmentation, while images (c) and (d) display the projection of the same regions.

Otsu's threshold has identified 91 objects in total, 12 of each can be considered significant based on their size. In regards to the validation of the known voids, the threshold was not able to fully separate any of them as single objects. The algorithm could outline the rough shape of the voids, but in both cases, the object remained connected to the rest of the foreground, forming one large connected object. This is primarily due to the characteristic of the voids of not being completely closed in all directions.

Sauvola's threshold was able to identify 731 objects in total, though most of them were quite small and therefore not considered as void candidates. After the filtering, 91 significant objects remained, including the Per-Tau shell. It is the method that has found the largest number of significant objects, although it is hard to know if they all represent real voids. As it is possible to see in the slices presented in Table 5.2, the Cepheus Flare shell is still connected to the biggest labeled object, which encompasses the whole space as part of the foreground. This shows the significant improvement of that can come from a local method in comparison to a global one such as Otsu's.

Region growing was not able to fully separate any object that could be considered a void candidate. Despite being able to delineate the void's shapes quite similarly to Otsu and Sauvola threshold, the foreground is all connected into one single object. When

looking at the slices in table 5.2, the Per-Tau shell looks separated from the background when at first glance, but a close inspection in the projection shows the connection to the foreground. The Cepheus Flare shell is also connected to the same structure.

The Morphological Chan Vese method has identified 11 significant objects, out of a total of 14 total objects. However, similarly to the region growing result, neither of the known voids are separated from the biggest label that represent the foreground.

The k-Means clustering has found 91 objects, 11 of which have a significant size. Just like the Sauvola method, k-means was able to correctly identify the Per-Tau shell but not the Cepheus-Flare shell.

The deep learning method was able to identify 14 significant objects out of a total of 33, including both the Per-Tau shell and the Cepheus-Flare shell. Despite these interesting results, it is important to consider the extreme data reduction needed for the application of the method, which comes at the price of data loss.

The Void-Finder algorithm, like the deep learning method, was able to identify both validation voids. The main difference once again lies in the number of significant objects that it has found, which at 44 is relatively higher compared to the other methods, second only to Sauvola.

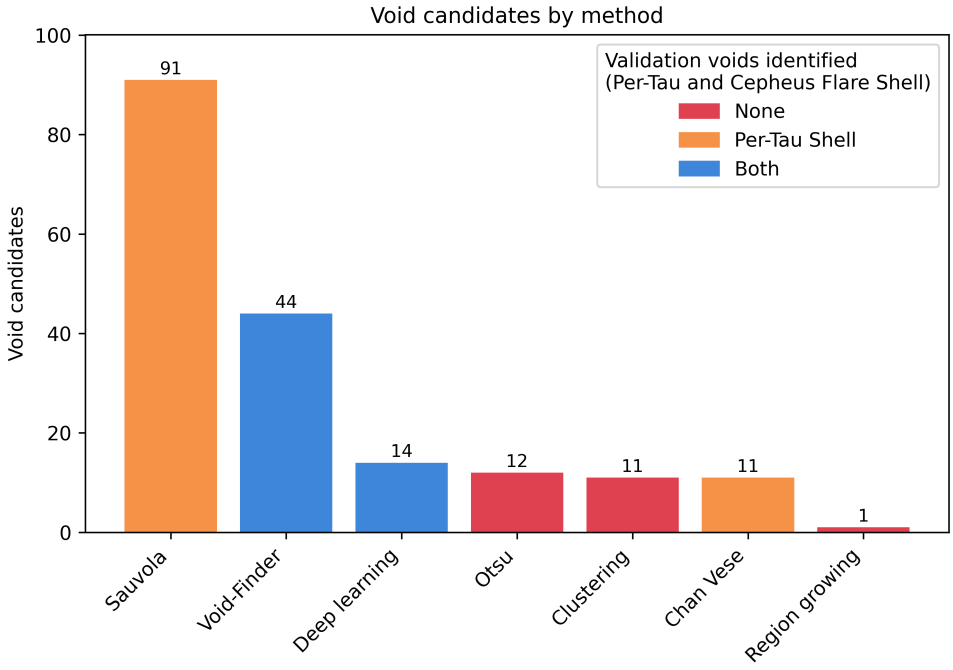


Figure 5.5.: Void candidates and validation voids identified by method for the dust map.

5.5. Runtime analysis

This section provides an overview of the runtime for the selected methods. The methods ran in a MacBook Air with a 1.8 GHz Dual-Code Intel Core i5 processor, 8GB of 1600MHz LPDDR3 onboard memory, 128GB of storage and Intel HD Graphics 6000 with no dedicated GPUs.

The Figure 5.6 shows the run times of the evaluated methods for the dust map data (740,740,540). Since the resources of the computer are rather limited, for the Chan-Vese and deep learning methods the data had to be reduced due to insufficient memory issues. The reduction of the data had a factor of 8x for the Chan Vese, and 27x for the deep learning method, resulting in input dimensions of (370,370,270) and (247,247,180), respectively. It is also important to note the run times presented are an average of multiple runs. Especially for the deep learning method and for the Void-Finder, the individual run times can vary a lot depending on the selected parameters and the convergence of the algorithm. For the Void-Finder, the run time displayed considers a run with 40960 initial seeds, averaging around 400 seed evaluations per minute.

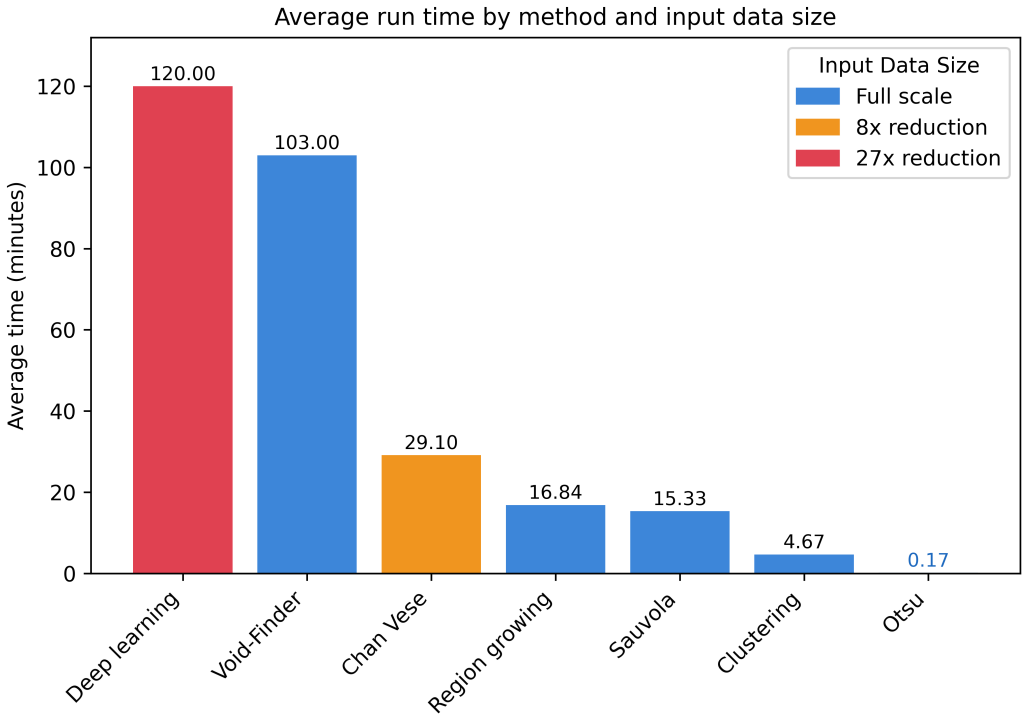


Figure 5.6.: Run times for the selected methods on the dust map.

6. Conclusion and future work

This thesis introduces the Void-Finder algorithm, a novel approach for detecting dust voids within the interstellar medium (ISM) using three-dimensional dust data. The challenge in identifying dust voids lies in the complex and irregular distribution of the ISM, which is influenced by various processes. This leads to voids with ill-defined boundaries and that can exist across different density levels.

To overcome this challenge, the Void-Finder leverages the radial density profile characteristic of supernova-induced voids, where the exploding star expels gas and dust from a central point, sweeping up material that forms an approximately spherical over-density around it. Beyond the main area affected by the explosion, the density gradually decreases again, returning to match the average density of the surrounding ISM. The Void-Finder assumes seed voxels in the dust map to be the center of a potential void, and then evaluates if the mean radial profile around that voxel matches the characteristic void pattern for classification. This approach allows the algorithm to identify voids even when low-density regions exist within the surrounding dust.

In comparison with established image segmentation techniques, the Void-Finder's primary strength lies in its focus on identifying and separating each void within the ISM, unlike traditional methods, which often struggle to fully separate adjacent voids when they are not completely enclosed or have low-density regions at their boundaries. While it may not achieve the highest accuracy in delineating the exact shape of voids (as reflected in lower Jaccard scores), it consistently outperforms other methods in isolating voids as distinct entities, a crucial capability for astrophysical research. Additionally, the algorithm shows robustness in locating void centers, even when the initial seeds are not precisely placed.

While promising, the Void-Finder is not without challenges. While it performs well across a broad set of parameters, selecting these parameters is not always intuitive and may require domain expertise. Furthermore, the algorithm requires some level of domain knowledge to filter out truly relevant voids based on desired characteristics. Another consideration is the computational cost. Depending on the number of seeds and precision needed, the algorithm can be time-consuming.

The other image segmentation methods evaluated in this thesis — Otsu's threshold, Sauvola's threshold, region growing, morphological Chan-Vese, k-Means clustering, and unsupervised deep learning — each exhibited varying levels of effectiveness. The deep learning method and k-Means clustering show promising results, especially in identifying known voids such as the Per-Tau shell. However, these methods can struggle to fully separate voids when their boundaries are close together, leading to merged or incomplete segmentation. Sauvola's threshold, while segmenting a large number of significant objects that can be considered potential voids, also faced difficulties in accurately isolating

meaningful voids from the foreground. Despite the deep learning method being capable of identifying both the Per-Tau shell and the Cepheus-Flare shell, it struggles with high computational costs and currently cannot scale to larger datasets.

Overall, this work lays the groundwork for future advancements in the field, suggesting that automated methods like the Void-Finder could become essential tools for astrophysicists. The algorithm's performance in separating significant voids suggests a promising direction for further research, including the potential refinement of the algorithm to enhance its accuracy and applicability to even more complex datasets.

Bibliography

[1] R. Adams and L. Bischof. 'Seeded region growing'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.6 (1994), pp. 641–647. DOI: 10.1109/34.295913.

[2] Y. Alzahrani and B. Boufama. 'Biomedical Image Segmentation: A Survey'. In: *SN Computer Science* 2 (July 2021). DOI: 10.1007/s42979-021-00704-7.

[3] H. G. Arce et al. 'A bubbling nearby molecular cloud: complete shells in Perseus'. In: *The Astrophysical Journal* 742.2 (Sept. 2011), p. 105. DOI: 10.1088/0004-637X/742/2/105.

[4] S. Arens and G. Domik. 'A Survey of Transfer Functions Suitable for Volume Rendering'. In: *IEEE/ EG Symposium on Volume Graphics*. Ed. by R. Westermann and G. Kindlmann. The Eurographics Association, 2010. ISBN: 978-3-905674-23-1. DOI: 10.2312/VG/VG10/077-083.

[5] D. Arthur and S. Vassilvitskii. 'k-means++: the advantages of careful seeding'. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. ISBN: 9780898716245.

[6] S. Berg et al. 'ilastik: interactive machine learning for (bio)image analysis'. In: *Nature Methods* (Sept. 2019). ISSN: 1548-7105. DOI: 10.1038/s41592-019-0582-9.

[7] A. K. Bhandari et al. 'A local contrast fusion based 3D Otsu algorithm for multilevel image segmentation'. In: *IEEE/CAA Journal of Automatica Sinica* 7 (2020), pp. 200–213.

[8] S. Bialy et al. 'The Per-Tau Shell: A Giant Star-forming Spherical Shell Revealed by 3D Dust Observations'. In: *The Astrophysical Journal Letters* 919.1 (Sept. 2021), p. L5. DOI: 10.3847/2041-8213/ac1f95.

[9] Y. Boykov and G. Funka-Lea. 'Graph Cuts and Efficient ND Image Segmentation'. In: *International Journal of Computer Vision - IJCV* 70 (Nov. 2006), pp. 109–131. DOI: 10.1007/s11263-006-7934-5.

[10] J. F. Canny. 'A Computational Approach to Edge Detection'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8 (1986), pp. 679–698.

[11] V. Caselles et al. 'Geodesic active contours'. In: *Proceedings of IEEE International Conference on Computer Vision*. 1995, pp. 694–699. DOI: 10.1109/ICCV.1995.466871.

[12] T. Chan and L. Vese. 'Active contours without edges'. In: *IEEE Transactions on Image Processing* 10.2 (2001), pp. 266–277. DOI: 10.1109/83.902291.

[13] B.-Q. Chen et al. 'A three-dimensional extinction map of the Galactic anticentre from multiband photometry'. In: *Monthly Notices of the Royal Astronomical Society* 443 (2014), pp. 1192–1210.

[14] L. D. Cohen. 'On active contour models and balloons'. In: *CVGIP Image Underst.* 53 (1991), pp. 211–218.

[15] D. P. Cox and R. J. Reynolds. 'The local interstellar medium'. In: *Annual Review of Astronomy and Astrophysics* 25 (1987), pp. 303–344.

[16] M. M. Foley et al. 'A 3D View of Orion. I. Barnard's Loop'. In: *The Astrophysical Journal* 947.2 (Apr. 2023), p. 66.

[17] R. Gonzalez and R. Woods. *Digital Image Processing Global Edition*. Pearson Deutschland, 2017, pp. 768–770. ISBN: 9781292223049.

[18] S. P. Goodwin et al. 'The relative size of the Milky Way'. In: *The Observatory* 118 (Aug. 1998), pp. 201–208.

[19] N. Gordillo et al. 'State of the art survey on MRI brain tumor segmentation'. In: *Magnetic Resonance Imaging* 31.8 (2013), pp. 1426–1438. ISSN: 0730-725X. DOI: 10.1016/j.mri.2013.05.002.

[20] G. M. Green et al. 'A 3D Dust Map Based on Gaia, Pan-STARRS 1, and 2MASS'. In: *The Astrophysical Journal* 887 (2019).

[21] I. A. Grenier et al. 'CO observations of the Cepheus flare. I. Molecular clouds associated with a nearby bubble'. In: *Astrophysical Journal; (USA)* 347 (Dec. 1989). ISSN: 0004-637X. DOI: 10.1086/168112.

[22] R. Harb and P. Knobelreiter. 'InfoSeg: Unsupervised Semantic Image Segmentation with Mutual Information Maximization'. In: *German Conference on Pattern Recognition*. 2021.

[23] T. Head et al. *Scikit-optimize: Sequential model-based optimization in Python*. Accessed: 2024-06-03. 2018. URL: https://scikit-optimize.github.io/stable/modules/generated/skopt.gp_minimize.html.

[24] J. Hyun Cho et al. 'PiCIE: Unsupervised Semantic Segmentation using Invariance and Equivariance in Clustering'. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 16789–16799. DOI: 10.1109/CVPR46437.2021.01652.

[25] P. Jaccard. 'The distribution of the flora in the alpine zone'. In: *New Phytologist* 11.2 (1912). https://onlinelibrary.wiley.com/doi/pdfdirect/10.1111/j.1469-8137.1912.tb05611.x, pp. 37–50. DOI: 10.1111/j.1469-8137.1912.tb05611.x.

[26] X. Ji et al. 'Invariant Information Clustering for Unsupervised Image Classification and Segmentation'. In: Oct. 2019, pp. 9864–9873. DOI: 10.1109/ICCV.2019.00996.

[27] H. J. Johnson et al. *The ITK Software Guide*. Third. *In press*. Kitware, Inc. 2013. URL: http://www.itk.org/ItkSoftwareGuide.pdf.

[28] M. H. Jones et al. 'A study of the interstellar dust distribution in regions of low total column density'. In: *Monthly Notices of the Royal Astronomical Society* 277.4 (Dec. 1995), pp. 1587–1598. ISSN: 0035-8711. DOI: 10.1093/mnras/277.4.1587. eprint: https://academic.oup.com/mnras/article-pdf/277/4/1587/4301859/mnras277-1587.pdf.

[29] A. Kanezaki. 'Unsupervised Image Segmentation by Backpropagation'. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 1543–1547. DOI: 10.1109/ICASSP.2018.8462533.

[30] H. Karttunen et al. 'Fundamental Astronomy'. In: Springer Berlin Heidelberg, 2007. Chap. 15. ISBN: 9783540341444.

[31] H. Karttunen et al. 'Fundamental Astronomy'. In: Springer Berlin Heidelberg, 2007. Chap. 4. ISBN: 9783540341444.

[32] M. Kass et al. 'Snakes: Active contour models'. In: *International Journal of Computer Vision* 1 (1988), pp. 321–331.

[33] N. Khalid. 'Review on Region-Based Segmentation Using Watershed and Region Growing Techniques and their Applications in Different Fields'. In: *Journal La Multiapp* 3 (Nov. 2022), pp. 241–249. DOI: 10.37899/journallamultiapp.v3i5.714.

[34] W. Kim et al. 'Unsupervised Learning of Image Segmentation Based on Differentiable Feature Clustering'. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 8055–8068. DOI: 10.1109/tip.2020.3011269.

[35] J. Kniss et al. 'Multidimensional transfer functions for interactive volume rendering'. In: *IEEE Transactions on Visualization and Computer Graphics* 8.3 (2002), pp. 270–285. DOI: 10.1109/TVCG.2002.1021579.

[36] R. H. Leike et al. 'Resolving nearby dust clouds'. In: *Astronomy & Astrophysics* 639, A138 (July 2020). DOI: 10.1051/0004-6361/202038169. arXiv: 2004.06732 [astro-ph.GA].

[37] R. H. Leike and T. A. Enßlin. 'Charting nearby dust clouds using Gaia data only (Corrigendum)'. In: *Astronomy & Astrophysics* (2019).

[38] J. Ma and X. Cheng. 'Fast segmentation algorithm of PCB image using 2D OTSU improved by adaptive genetic algorithm and integral image'. In: *Journal of Real-Time Image Processing* 20 (2023), pp. 1–11.

[39] P. Márquez-Neila et al. 'A Morphological Approach to Curvature-Based Evolution of Curves and Surfaces'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.1 (2014), pp. 2–17. DOI: 10.1109/TPAMI.2013.106.

[40] D. Marr and E. Hildreth. 'Theory of Edge Detection'. In: *Proceedings of the Royal Society of London. Series B, Containing papers of a Biological character. Royal Society (Great Britain)* 207 (Feb. 1980), pp. 187–217. DOI: 10.1098/rspb.1980.0020.

[41] D. Marshall et al. 'Modelling the Galactic interstellar extinction distribution in three dimensions'. In: *http://dx.doi.org/10.1051/0004-6361:20053842* 453 (Apr. 2006). DOI: 10.1051/0004-6361:20053842.

[42] T. McInerney and D. Terzopoulos. 'Topologically adaptable snakes'. In: *Proceedings of IEEE International Conference on Computer Vision*. 1995, pp. 840–845. DOI: 10.1109/ICCV.1995.466850.

[43] P. Mesejo et al. 'A survey on image segmentation using metaheuristic-based deformable models: state of the art and critical analysis'. In: *Applied Soft Computing* 44 (2016), pp. 1–29. ISSN: 1568-4946. DOI: 10.1016/j.asoc.2016.03.004.

[44] S. E. Mirsadeghi et al. 'Unsupervised Image Segmentation by Mutual Information Maximization and Adversarial Regularization'. In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6931–6938. DOI: 10.1109/LRA.2021.3095311.

[45] W. Niblack. 'An introduction to digital image processing'. In: 1986.

[46] C. A. Olano et al. 'The interstellar medium in the Upper Cepheus-Cassiopeia region'. In: *Monthly Notices of the Royal Astronomical Society* 369 (2006), pp. 867–874.

[47] Y. Ouali et al. 'Autoregressive Unsupervised Image Segmentation'. In: *ArXiv* abs/2007.08247 (2020).

[48] D. S. Prabha and J. S. Kumar. 'Performance analysis of image smoothing methods for low level of distortion'. In: *2016 IEEE International Conference on Advances in Computer Applications (ICACA)*. 2016, pp. 372–376. DOI: 10.1109/ICACA.2016.7887983.

[49] S. E. Sale et al. 'A 3D extinction map of the northern Galactic plane based on IPHAS photometry'. In: *Monthly Notices of the Royal Astronomical Society* 443 (2014), pp. 2907–2922.

[50] J. Sauvola and M. Pietikäinen. 'Adaptive document image binarization'. In: *Pattern Recognition* 33.2 (2000), pp. 225–236. ISSN: 0031-3203. DOI: 10.1016/S0031-3203(99)00055-2.

[51] A. Sekmen et al. 'Unsupervised deep learning for subspace clustering'. In: *2017 IEEE International Conference on Big Data (Big Data)*. 2017, pp. 2089–2094. DOI: 10.1109/BigData.2017.8258156.

[52] M. Sezgin and B. Sankur. 'Survey over image thresholding techniques and quantitative performance evaluation'. In: *Journal of Electronic Imaging* 13 (Jan. 2004), pp. 146–168. DOI: 10.1117/1.1631315.

[53] A. Shafi and D. Padha. 'Medical Image Segmentation A Review of Recent Techniques, Advancements and a Comprehensive Comparison'. In: *International Journal of Computer Sciences and Engineering* 7 (July 2019), pp. 114–124. DOI: 10.26438/ijcse/v7i7.114124.

[54] J. Shi and J. Malik. 'Normalized Cuts and Image Segmentation'. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22.8 (Aug. 2000), pp. 888–905. ISSN: 0162-8828. DOI: 10.1109/34.868688.

[55] I. M. Sobol. 'On the distribution of points in a cube and the approximate evaluation of integrals'. In: *USSR Computational Mathematics and Mathematical Physics* 7 (1967), pp. 86–112.

[56] H. Sponton and J. Cardelino. 'A Review of Classic Edge Detectors'. In: *Image Process. Line* 5 (2015), pp. 90–123.

[57] P. Sthitpattanapongsa and T. Srinark. 'An Equivalent 3D Otsu's Thresholding Method'. In: *Pacific-Rim Symposium on Image and Video Technology*. 2011.

[58] K. Sum and P. Y. Cheung. 'Boundary vector field for parametric active contours'. In: *Pattern Recognition* 40.6 (2007), pp. 1635–1645. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2006.11.006.

[59] R. Sun et al. 'Survey of Image Edge Detection'. In: *Frontiers in Signal Processing*. 2022.

[60] M. Szilágyi et al. 'The *Gaia* view of the Cepheus OB2 association'. In: *Monthly Notices of the Royal Astronomical Society* 520.1 (Jan. 2023), pp. 1390–1410. DOI: 10.1093/mnras/stad027.

[61] N. Tajbakhsh et al. 'Embracing Imperfect Datasets: A Review of Deep Learning Solutions for Medical Image Segmentation'. In: *Medical Image Analysis* 63 (Apr. 2020), p. 101693. DOI: 10.1016/j.media.2020.101693.

[62] S. Tan et al. 'Adaptive region-growing with maximum curvature strategy for tumor segmentation in 18F-FDG PET'. In: *Physics in Medicine & Biology* 62.13 (June 2017), p. 5383. DOI: 10.1088/1361-6560/aa6e20.

[63] A. Wadhwa et al. 'A review on brain tumor segmentation of MRI images'. In: *Magnetic Resonance Imaging* 61 (2019), pp. 247–259. ISSN: 0730-725X. DOI: 10.1016/j.mri.2019.05.043.

[64] S. Walch et al. 'The SILCC (SImulating the LifeCycle of molecular Clouds) project: I. Chemical evolution of the supernova-driven ISM'. In: *Monthly Notices of the Royal Astronomical Society* 454 (Dec. 2014). DOI: 10.1093/mnras/stv1975.

[65] B. Welsh et al. 'EUV mapping of the local interstellar medium: The Local Chimney revealed?' In: *Astronomy and Astrophysics* 352 (Nov. 1999), pp. 308–316.

[66] O. Wirjadi. *Survey of 3D image segmentation methods*. Tech. rep. 123. Fraunhofer (ITWM), 2007. URL: http://nbn-resolving.de/urn:nbn:de:hbz:386-kluedo-15457.

[67] X. Xia and B. Kulis. 'W-Net: A Deep Model for Fully Unsupervised Image Segmentation'. In: (Nov. 2017). DOI: 10.48550/arXiv.1711.08506.

[68] J. Xing et al. 'Robust 2D Otsu's Algorithm for Uneven Illumination Image Segmentation'. In: *Computational Intelligence and Neuroscience* 2020 (2020).

[69] C. Xu and J. Prince. 'Gradient vector flow: a new external force for snakes'. In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1997, pp. 66–71. DOI: 10.1109/CVPR.1997.609299.

[70] T. S. Yoo et al. 'Engineering and algorithm design for an image processing API: A technical report on ITK–the Insight Toolkit'. English. In: *Studies in Health Technology and Informatics* 85 (2002), pp. 586–592. ISSN: 0926-9630.

[71] W. Zhou et al. 'Techniques for Image Segmentation Based on Edge Detection'. In: *2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI)*. 2021, pp. 400–403. DOI: 10.1109/CEI52496.2021.9574569.

[72] Z. Zhou and X.-J. Wen. 'A Survey of Segmentation Algorithms Based on ITK'. In: vol. 30. 2019, pp. 305–313.

[73] C. Zucker et al. 'Star formation near the Sun is driven by expansion of the Local Bubble'. In: *Nature* 601.7893 (Jan. 2022), pp. 334–337. ISSN: 1476-4687. DOI: 10.1038/s41586-021-04286-5.

[74] C. Zucker et al. *The Solar Neighborhood in the Age of Gaia*. 2023. arXiv: 2212.00067 [astro-ph.GA].

A. Data preprocessing

This section details the methods applied to the raw data as well as the reasoning behind the selected method and parameters.

A.1. Selected methods

The split of the original data into octants for parallelization, done by the authors of the dust map, created inconsistencies in the data range across different sections of the map. To improve the cohesion of these parts, several preprocessing approaches were considered and tested in the dust map.

Since the main issue occurs at the borders where the octants meet, local preprocessing methods were preferred to maintain the overall balance within the octants while smoothing the transitions between them. Methods such as the Gaussian filter and median filter are popular choices for smoothing images based on local information and can help soften the octant transitions. Anisotropic diffusion is another approach that aims to preserve edges where possible and was also tested for this purpose.

To evaluate the preprocessing of the data, metrics such as Peak Signal-to-Noise Ratio (PSNR) and the segmentation volume of the known voids were compared. The main idea behind these metrics is to identify a range of parameters that improve the octant transitions, while maintaining or improving the segmentation of known void areas and avoiding overprocessing of the data.

Peak Signal-to-Noise Ratio (PSNR) is a widely used metric for evaluating the quality of an image by quantifying the ratio between the maximum possible power of a signal (image) and the power of noise affecting its fidelity [48]. Higher PSNR values indicate better image quality, as they reflect lower levels of distortion or noise. In image preprocessing, PSNR is often employed to assess the effectiveness of denoising, compression, or enhancement techniques by comparing the processed image to the original. For this comparison, the PSNR was calculated between the original image and the preprocessed result to quantify the impact of preprocessing relative to no processing at all.

The Gaussian filter is a linear spatial filter used to reduce noise and detail in an image by convolving it with a Gaussian function. A Gaussian kernel is applied to the image, where each voxel's new value is a weighted average of its neighbors, with closer neighbors receiving higher weights. The standard deviation (σ) of the Gaussian function determines the level of smoothing [48].

Median filtering is a non-linear method for noise reduction, where each voxel in a 3D image is replaced with the median value of the voxels in a surrounding neighborhood based on a kernel of size s. This filter tends to preserve edges better than linear filters

[48].

Anisotropic diffusion, also called Perona-Malik filtering, reduces image noise while preserving edges by allowing diffusion only within regions of similar intensity. This method solves a partial differential equation where diffusion coefficients are adjusted based on the local gradient magnitude. While the main advantage of this method is the preservation of edges, it may struggle with high gradient fluctuations around the edges [48]. For the dust map data, an implementation of gradient anisotropic diffusion from the Insight Toolkit (ITK) was used. The method is controlled by three primary parameters: number of iterations, time step, and conductance. According to the ITK guide [27], the appropriate number of iterations depends on the input image. The conductance is a parameter that controls the sensitivity of the diffusion process; typically, higher values result in more diffuse image features. The time step parameter sets the time increment for the process. While large timesteps can accelerate diffusion, too large a timestep can lead to instability in the algorithm. The recommended time step for a 3D image is to be at or below 0.0625 [27].

A.1.1. Pre-processing evaluations

Comparing the Peak Signal-to-Noise Ratio (PSNR) and its derivative for both the Gaussian and median filters with different parameters (standard deviation σ for the Gaussian filter and kernel size s for the median filter) shows the strength and rate of change of the parameters. While increasing the intensity of the filters results in significant PSNR changes for the first two levels, starting from a standard deviation and kernel size of 3, the impact of the filters on the PSNR becomes less pronounced, as shown in Figure A.1. It is also evident that at the initial levels (1 and 2), both methods have similar effects on the image. However, as the standard deviation and kernel size increase, the Gaussian filter has a stronger smoothing effect on the image than the median filter, as indicated by the lower PSNR.

For anisotropic diffusion, the PSNR was calculated to assess the effects of the number of iterations and the conductance parameter. The time step was kept at the suggested value of 0.0625 for 3D images, as slower time steps would require more iterations and be less time-efficient. In Figure A.2, we can see that starting from 15 iterations, the changes in PSNR begin to stabilize. It is also apparent that the biggest difference in PSNR for the conductance parameter occurs between values 1 and 2, with higher conductance values resulting in similar smoothing effects.

Figure A.3 shows the projection of the gradient magnitude of the original dust map data and the preprocessed data with different filter parameters. For anisotropic diffusion, the conductance is fixed at 5 for a stronger smoothing effect, while the number of iterations varies. All filters show improvements in the overall range of the full map compared to the original. However, the borders where the octants meet still display prominent gradients, with unnatural lines across the image.

To verify the effects of the filters on void segmentation, Otsu's threshold (detailed in Chapter 3) was applied to the preprocessed data in the Per-Tau Shell region, and the variation in the resulting volume was compared. Figure A.4 shows the progression of the

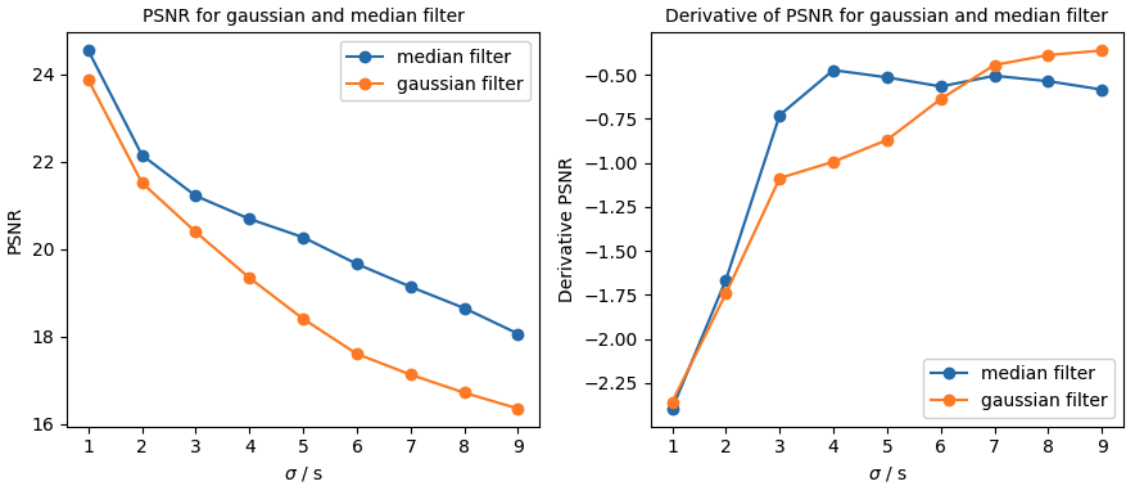


Figure A.1.: PSNR variation for Gaussian and median filter

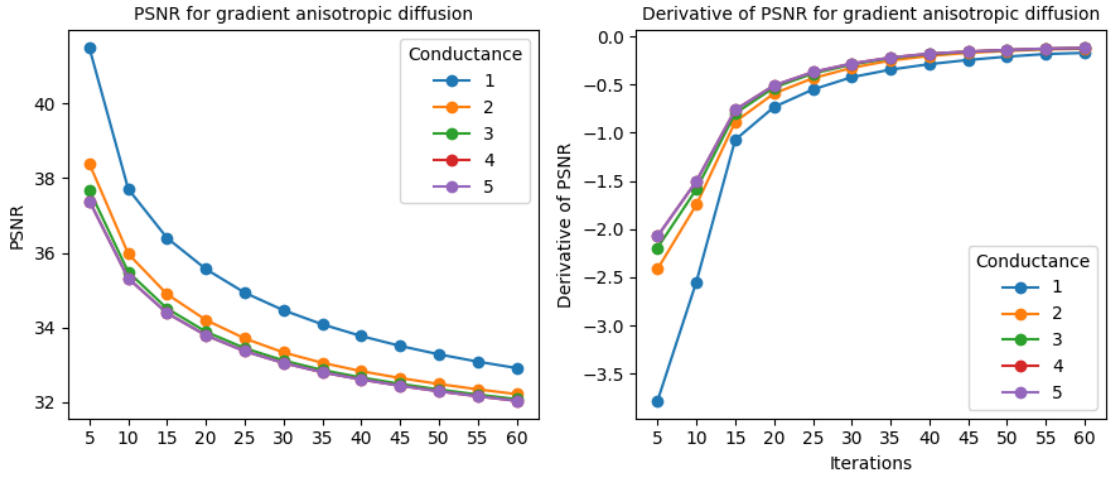


Figure A.2.: PSNR variation for gradient anisotropic diffusion at different iterations and conductance.

volume as the parameters change for the selected methods. The segmentation slices are shown in Figure A.5.

Based on the overall metrics and visual analysis, the median filter with a kernel size of $s = 4$ was selected for preprocessing. Although anisotropic diffusion demonstrated good performance in uniformizing the gradient range and maintaining a higher PSNR, the line formed at the octant borders was more pronounced compared to the median and Gaussian filters. While both the Gaussian and median filters improved the transitions between octants, the median filter provided better smoothing while preserving finer image details.

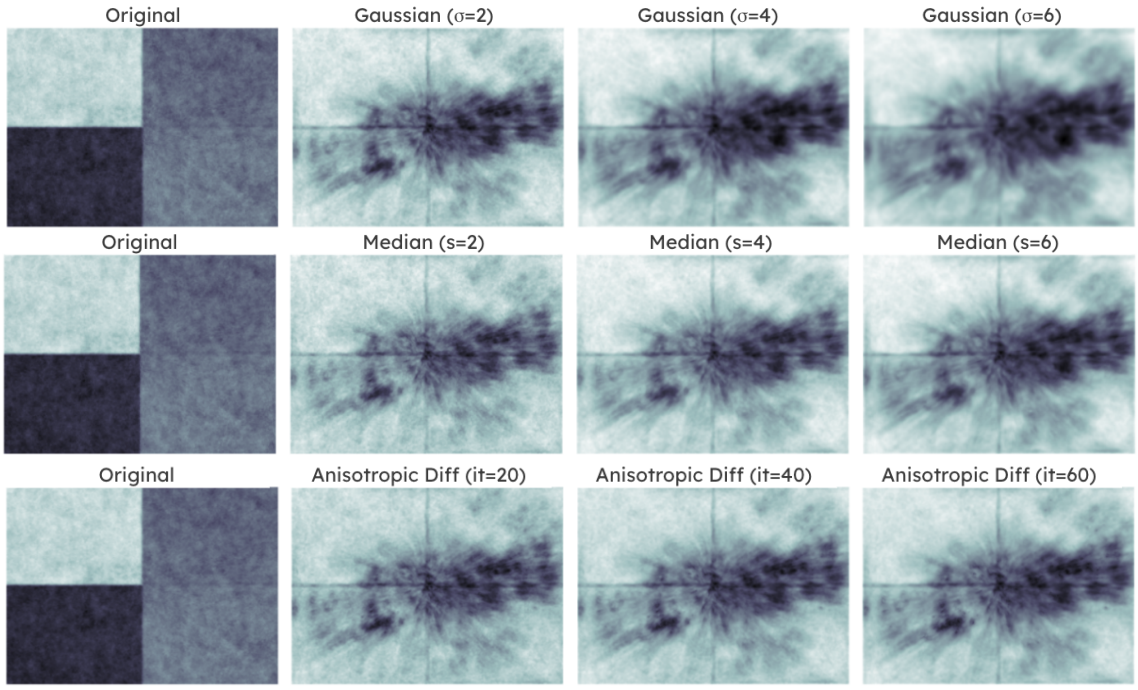


Figure A.3.: Gradient magnitude projection over different parameters for the Gaussian filter, median filter and anisotropic diffusion.

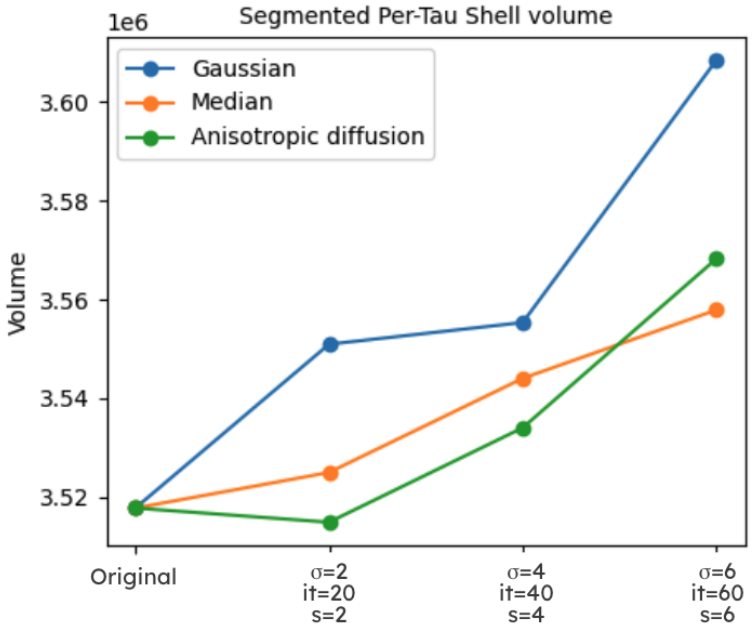


Figure A.4.: Volume variation for the segmentation in the Per-Tau Shell region with the different preprocessing filters.

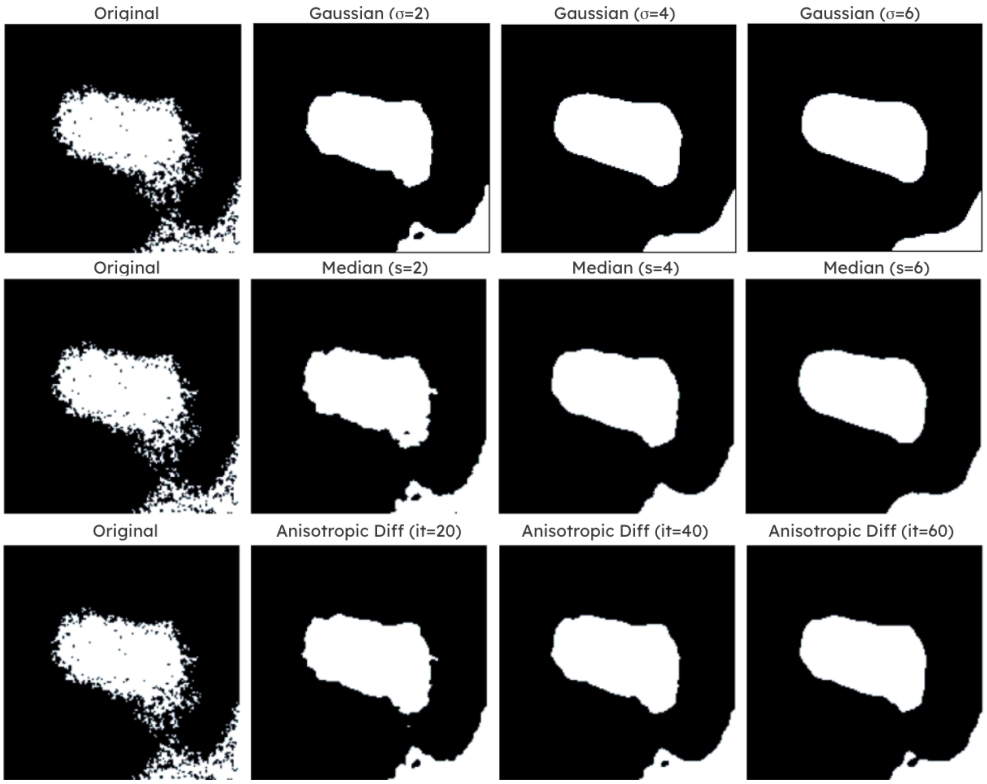


Figure A.5.: Otsu's segmentation of the Per-Tau Shell region over different parameters for the Gaussian filter, median filter and anisotropic diffusion.