

École polytechnique de Louvain

Semi-Automated Detection of sEEG Electrodes in Computed Tomography Scans

Author: Quentin DE LAET

Supervisors: Benoît MACQ, Ron KUPERS

Readers: Tim BARY, Hélène VERHAEGHE

Academic year 2024–2025

Master [120] in Computer Science and Engineering

Abstract

Stereoelectroencephalography (sEEG) is a critical procedure in the pre-surgical evaluation of patients with drug-resistant epilepsy, allowing the identification of the epileptogenic zone (EZ) through intracranial recordings [1], [2]. However, localizing the implanted electrode contacts in post-operative computed tomography (CT) scans remains a time-consuming and error-prone task, especially when dealing with hundreds of contacts per patient. This thesis proposes an algorithm for the accurate and efficient localization and labeling of sEEG electrode contacts in CT scans.

This work leverages mathematical and computational approaches that have not been previously explored in the context of sEEG electrode localization. Mathematical morphology [3], [4] is used to extract punctual information about the contacts and detect intersecting electrodes. Notions of graph theory [5], [6] and graph search procedures [7] are then applied to retrieve optimal linear models for the electrodes. A final post-processing step is performed to account for electrode bending and ensure coherent spacing between the contacts identified.

The pipeline is validated on clinical CT data from patients undergoing sEEG at Rigshospitalet, Copenhagen. Results demonstrate promising accuracy and high robustness to CT artifacts and electrode bending.

The code produced for this thesis can be found in the following repository: https://github.com/Quent-DL/MasterThesis_ElectroLoc.

Keywords — stereoelectroencephalography (sEEG), electrode localization, computed tomography (CT), mathematical morphology, graph theory, epilepsy

Acknowledgements

I am deeply thankful to **Benoît Macq**, who offered me the opportunity to work on this master's thesis and discover the fascinating field of medical informatics. As a result of this project, I developed a strong interest in research on epilepsy treatments, and I intend to pursue further work in this direction.

I would like to sincerely thank **Ron Kupers** and **Laurence Dricot** for their interest in the project, and their kindness throughout the year. Our regular meetings helped me gain a deeper understanding of medical imaging and neuroscience.

I am especially grateful to **Nicolas Delinte** and **Tim Bary** for their constant support, from the earliest stages of this master's thesis to its very end. Their unparalleled availability—as well as their feedback, suggestions, and expertise—constituted valuable help in the completion of this project, and saved me countless hours of struggle.

I would also like to thank **Hélène Verhaeghe** for kindly agreeing to read and evaluate this paper, and for showing interest in the outcome of months of passionate research and programming.

Special thanks are due to **Quentin Dessain** and **Vincent Joris** for their expertise and insightful responses to challenges encountered during the course of this thesis.

Finally, my utmost gratitude goes to **my parents**, for the constant and unconditional encouragement they have offered me throughout my life, and that they continue to offer every day. Their unfaltering support in helping me develop my interests and pursue my ambitions has led me to this culminating point in my studies: the completion of my master's thesis and, with it, my master's degree.

Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

I acknowledge the use of AI tools (specifically ChatGPT) to assist with the academic formulation of my ideas, limited to the abstract, introduction, and conclusion of this thesis. All core content, including the research, implementation, analysis, and writing, remains entirely my own.

Contents

Abstract	iii
Acknowledgements	v
Contents	viii
List of Figures	x
List of Tables	xi
Acronyms	xiii
Introduction	1
 I Context and State of the Art	 3
1 Introduction to Computed Tomography	5
1.1 X-Rays, Attenuation and Radiography	5
1.2 Computed Tomography: From 2D to 3D	7
1.3 World and Image Coordinate Systems	8
1.4 Relevant Artifacts in CT Scans	10
2 Stereoelectroencephalography	13
2.1 Basics of Stereoelectroencephalography	13
2.2 Locating Contacts and Existing Tools	16
3 Mathematical Morphology	19
3.1 Basics of Mathematical Morphology and Binary Images	19
3.2 Dilation and Erosion	20
3.3 Image Reconstruction by Dilation	20
3.4 Ultimate Erosion	21
4 Graph Theory and Search Algorithms	25
4.1 Basics of Graph Theory	25
4.2 Graph Searching with Breadth-First Search	27

II	Pipeline for Semi-Automated Detection of Electrode Contacts	29
5	Materials	31
5.1	Data Acquisition	31
5.2	Preprocessing	32
6	Methods	35
6.1	Extraction of Meaningful Points	35
6.2	Linear Modeling of the Electrodes	43
6.3	Post-Processing the Models	51
6.4	Algorithm Summary	56
6.5	Performance Assessment Metrics	57
7	Results and Discussions	61
7.1	Performances of the Algorithm	61
7.2	Comparison to Existing Methods	63
7.3	Limitations	64
7.4	Perspectives and Potential Improvements	66
	Conclusion	69
	References	71
	Appendices	75
A	Code and Repository Overview	77
A.1	General Information about the Project	77
B	Time-Parametrized Parabolic Curves	79
B.1	Definition of 3-Dimensional Parabolas	79
B.2	Fitting a Parabola to a Set of Points	80
B.3	Ignoring Outliers	82
B.4	Summary of the Process	82

List of Figures

1.1	The spectrum of electromagnetic waves	6
1.2	Sources of attenuation of X-rays as a function of energy	6
1.3	Illustration of the radiography process	7
1.4	Example of radiograph	7
1.5	One point expressed in two different 2D coordinate spaces.	9
1.6	A slice of CT scan with strong metal artifacts	11
2.1	Illustration of the general concept of stereoelectroencephalography	14
2.2	Components of a depth electrode	14
2.3	A slice of post-surgical CT scan	15
3.1	Example of binary image.	20
3.2	Example of dilation and erosion	20
3.3	Example of image reconstruction	21
3.4	Example of the concept of object in a binary image	22
3.5	Example of ultimate erosion	22
4.1	Examples of weighted and unweighted graphs	25
4.2	Example of application of Breadth-First Search	27
5.1	A 3D view of CT volume with electrode cables	33
5.2	A slice of CT scan, and the transparent brain mask	33
6.1	CT histograms before and after applying the brain mask	36
6.2	A quantized slice of CT scan to highlight ranges of values	37
6.3	A mask for electrodes, viewed both in 2D and 3D	37
6.4	The colored connected components in an electrode mask	38
6.5	3D view of the colored post-dilation connected components of electrodes	39
6.6	3D view of the output of ultimate erosion with a 3D cross	40
6.7	3D view of the centers of the connected components in figure 6.6	40
6.8	Comparison between a 3D cross and a masked electrode	41
6.9	Bar plots of ultimate erosion's performance for several structuring elements	41
6.10	Result of ultimate erosion with an in-slice cross	42
6.11	Summary of the centroid extraction module	42
6.12	Kernel Density Estimation used for the computation of inter-contact distance	44
6.13	3D view of the minimum spanning trees computed from the centroids	45
6.14	Examples of electrode trees and their points of interest	46
6.15	Example of search graph used to compute optimal pairs of centroids	47
6.16	Effect on the search graph of pruning nodes in BFS	48

6.17	Definition of the line segment models	49
6.18	Distance between a point and a segment model	50
6.19	The weight of a centroid for evaluating a node in the search graph	50
6.20	Summary of the linear modeling module	51
6.21	A simple example of time-parametrized parabolic curve	53
6.22	Distance between a point and a parabola	53
6.23	Replacement of a straight model by a curved one	54
6.24	Illustration of stable matching	54
6.25	Summary of the post-processing module	56
6.26	Summary of the overall algorithm	57
6.27	Examples of matched predicted and expected contacts	59
7.1	Effect of the dilation radius on the algorithm's performance	61
7.2	3D view of the output of the linear modeling module	62
7.3	Performance of linear modeling as a function of the branching factor	62
7.4	Density function of the error between expected and predicted contacts	63
7.5	3D view of the output of the algorithm, with false positives and false negatives	64
7.6	Presence of a contact-less part of shaft in a CT image	65
7.7	Examples of curves handled poorly by parabolas	66
7.8	Incorrect fit of the linear models due to wrong identification of DCCs	66
7.9	The effect of hyperparameter R on the number of DCCs detected	67
7.10	Early prototype of GUI for the algorithm	68
B.1	Example of parabolic curve	80
B.2	Assumption about the projection time of the samples when fitting a parabola	81

List of Tables

1.1	Typical values of Hounsfield Units in CT scans	8
2.1	Comparison between CT and MRI for post-implantation evaluation	15
2.2	Summary of the existing tools for locating contacts	18
5.1	Resolutions of the dataset's CT images	31
6.1	Inputs and outputs of the centroid extraction module	43
6.2	Inputs and outputs of the modeling module	51
6.3	Information required by the contact generation module	54
6.4	Inputs and outputs of the post-processing module	57
6.5	Confusion matrix used for the assessment of the algorithm	58
7.1	Performances of the algorithm	63
7.2	Comparison between existing tools and this algorithm	65
A.1	Format of the CSV required by the algorithm	77
A.2	Format of the optional ground truth to provide to the algorithm	78
A.3	Format of the algorithm's output CSV	78

Acronyms

Acronym	Definition
BFS	Breadth-First Search
CBCT	Cone-Beam CT
CC	Connected Component
CSV	Comma-Separated Values
CT	Computed Tomography
DCC	Post-Dilation Connected Component
DCCid	Post-Dilation Connected Component Identifier
DICOM	Digital Imaging and Communication in Medicine
EEG	Electroencephalogram
EID	Electrode Identifier
EP	Entry Point
EZ	Epileptogenic Zone
GTC	Ground Truth Contact
GUI	Graphical User Interface
HU	Hounsfield Units
MDCT	Multi-Detector CT
MM	Mathematical Morphology
MNI	Montreal Neurological Institute
MRI	Magnetic Resonance Imaging
NIFTI	Neuroimaging Informatics Technology Initiative
PC	Predicted Contact
PID	Postional Identifier
POI	Point of Interest
PPV	Positive Predictive Value
sEEG	Stereoelectroencephalography

Introduction

Epilepsy is a neurological disorder marked by a persistent predisposition to generate seizures, affecting approximately 50 million people worldwide [8]. Among this population, around one-third suffers a form of epilepsy that is refractory to medical treatment. An alternative cure for these patients is the application of laser interstitial thermotherapy, a non-invasive surgery technique that ablates the patient’s epileptogenic zone (EZ) [9]. The success of this surgery critically depends on the precise identification and localization of the EZ.

When non-invasive methods such as electroencephalography (EEG) or structural and functional magnetic resonance imaging (MRI) fail to localize the EZ, clinicians may turn to invasive approaches like stereoelectroencephalography (sEEG) [1], [2]. SEEG involves the stereotactic implantation of depth electrodes, each containing between 5 and 18 metallic contacts, to record the patient’s brain electrical activity over a few days [10]. This technique is also employed in research for cognitive neuroscience and functional mapping due to its ability to reach deep brain structures and deliver high signal-to-noise ratio [11].

Accurate placement and post-operative localization of these electrode contacts is essential for both clinical and research purposes. In clinical contexts, the relationship between the signals measured and their location in the patient’s brain is central to the decisions made regarding resective surgery. From a research perspective, precise localization and conversion to a standardized space, such as the Montreal Neurological Institute (MNI) [12], is necessary to facilitate group-level analysis and integration with brain atlases.

For that purpose, post-operative imaging, typically via computed tomography (CT), is used to assess whether electrodes were correctly implanted. However, the manual segmentation and localization of electrode contacts is a time-consuming and error-prone process, particularly in cases involving over 200 contacts or complex spatial configurations. Other challenges such as CT artifacts or vertically-oriented electrodes can make the task significantly more complex.

The demand for accurate and efficient approaches to sEEG electrode localization has therefore grown significantly. While several automated and semi-automated methods have been proposed to ease the process of electrode contact segmentation and labeling [13]–[21], they rely on essential hypotheses about the trajectory of the electrodes or the severity of the image artifacts.

This thesis addresses the critical need for improved methods in SEEG electrode contact segmentation and localization. By developing or enhancing automated approaches, it aims to reduce clinicians’ workload, increase reproducibility, and ultimately support better surgical outcomes for patients with drug-resistant epilepsy.

Part I

Context and State of the Art

Chapter 1

Introduction to Computed Tomography

This first chapter focuses on the physical principles behind the Computed Tomography (CT) medical imaging technique, and how it is used in modern medicine, as well as its qualities and limitations.

This chapter starts by explaining the physical principles behind CT imaging. It then details the content of a CT scan and how this information is localized in real-world space. It ends with a presentation of relevant challenges regarding the processing of CT scans, in the form of imaging artifacts.

1.1 X-Rays, Attenuation and Radiography

CT is an imaging technique developed during the 1960s, which measures structural information about the object of interest. It is used nowadays in both industrial and medical contexts [22]. CT scans heavily rely on the concepts of X-rays, attenuation and radiography, which are briefly presented in this first section.

There are two fundamental types of waves: mechanical and electromagnetic (EM). This section focuses on the latter. Contrary to mechanical waves, EM waves can propagate through space without any material medium [23]. Each individual EM wave is called a photon and carries a certain quantity E of energy. In 1900, Max Planck proposed the following relation:

$$E(f) = hf \quad [eV], \quad (1.1)$$

which expresses the quantity E of energy carried by an EM wave as a function of its frequency of oscillation f [Hz], where $h \approx 6.626 * 10^{-34} [J \cdot s]$ is Planck's constant. The energy is expressed in eV (electron volts), where $1 eV := 1.6 * 10^{-19} [J]$. Following the electromagnetic radiation equation, the value f can itself be expressed as a function of the wave's velocity v [m/s] and wavelength λ [m]:

$$f = \frac{v}{\lambda} \quad \Rightarrow \quad E \propto \frac{1}{\lambda}. \quad (1.2)$$

The spectrum of EM waves has been split into multiple categories based on frequency (*i.e.* wavelength, or energy in the void), though the exact range of frequencies of each category is not standard and can vary between sources. Figure 1.1 presents this spectrum, which spans from

radio waves (least energetic) to γ -rays (most energetic). Among the spectrum also lie visible light and X-rays. Though the exact energy boundaries for those vary, academic sources usually agree that X-rays range from approximately 10 keV to 100 keV.

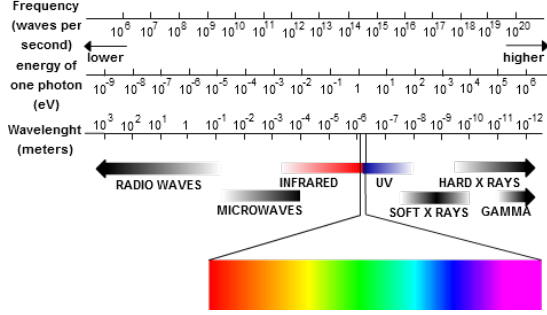


Figure 1.1: The spectrum of EM waves. The boundaries between categories are not standard and can vary between sources. Soft X-rays range from 10 keV to 100 keV approximately [24].

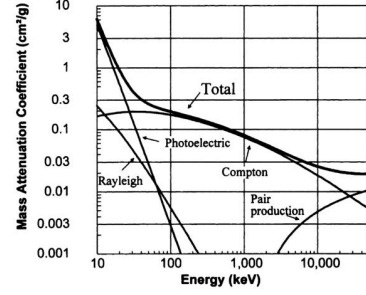


Figure 1.2: Rayleigh scattering, Compton scattering, photoelectric absorption and pair production and total mass attenuation coefficients as a function of radiation energy, for $Z \approx 7.5$ (approximation of soft tissue in the human body). The aggregation of these four interactions is the total attenuation of the material [23].

When an EM radiation is sent through matter, some photons interact with atoms on their path, leading to a loss of energy (*i.e.* absorption) and/or a change of trajectory (*i.e.* scattering). Without delving into the details, the four types of interactions are: Rayleigh scattering, Compton scattering, photoelectric absorption, and pair production [25]. Consequently, the number of photons that emerge from the far side of an irradiated object is lower than the number of photons that originally entered it. The aggregation of the four interaction types is called attenuation, and is intrinsic to the material traversed by the photons. Materials with low attenuation let photons traverse them more easily. Attenuation is quantified by the linear attenuation coefficient $\mu [m^{-1}]$, also denoted as $\mu(E)$, which depends on [26]:

- the mass density $\rho [g/m^3]$ of the material, which itself depends on its atomic number Z ;
- the energy E of the photon traversing the material.

Sometimes, the mass attenuation coefficient μ/ρ is used instead. Figure 1.2 depicts the change in μ/ρ as a function of the traversing photon's energy E . It shows that low-energy X-rays (≈ 10 keV) mainly undergo photoelectric absorption, whereas high-energy X-rays (≈ 100 keV) are mainly affected by Compton scattering. Overall, high-energy photons tend to get less attenuated than low-energy ones.

When radiating an object of thickness $x [m]$ with a monochromatic ray of photons with energy E , the Beer-Lambert law describes the ratio between the number of photons I_0 entering one end, and the number of photons I exiting through the other [27]:

$$I = I_0 e^{-x\mu(E)} \quad (1.3)$$

Radiography is a non-invasive imaging technique used to capture the differences in attenuation in regions of a patient's body. It works by emitting bremsstrahlung X-rays directed at the patient. The term "bremsstrahlung" refers to the electromagnetic radiation emitted when a fast-moving electron is decelerated or deflected by the electric field of an atomic nucleus, causing it to lose kinetic energy and release a photon, typically in the X-ray range [28], [29]. These X-rays are polychromatic (*i.e.* with various energies) and produced continuously by a source.

Photons traversing different regions of the body encounter on their path different tissue, with varying thickness and density. Hence, following the Beer-Lambert law (equation 1.3), they get attenuated with different strength. The intensity of photons emerging from the patient's body is then measured by a detector for each trajectory.

Figure 1.3 illustrates the process of radiography. The differences in intensity between the emitted and detected radiation for all trajectories are then processed by a computer. The result is displayed as a grayscale 2D image, called radiograph, in which each pixel represents a projection of the thickness and linear attenuation coefficient of the different tissue encountered along a trajectory [26]. Darker pixels represent lower thickness and/or attenuation. Figure 1.4 shows an example of radiograph.

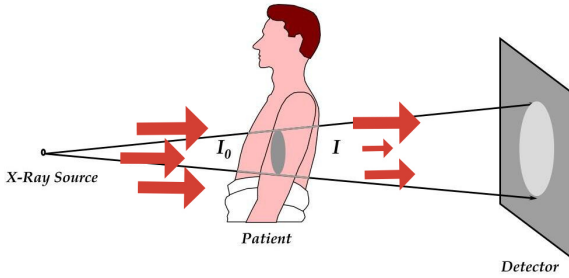


Figure 1.3: *Illustration of the radiography process. A radiation with intensity I_0 is sent through the patient. The intensity of the photons that fully traverse the body is measured by a detector, and used to produce a radiograph. Adapted from [30].*



Figure 1.4: *A radiograph of a patient's chest. Bright pixels represent high attenuation and/or thickness [31].*

1.2 Computed Tomography: From 2D to 3D

The previous subsection detailed the physical principles behind radiography. But such imaging technique possesses a strong weakness; while the linear attenuation coefficient of the patient's tissue is a 3-dimensional function that depends on position in space, radiography is only capable of representing a 2-dimensional projection of that function. This means that information is lost during the process.

CT makes up for that weakness by using numerous 2D radiographs, each taken from a different angle around the patient. Over the years, several generations of CT were developed, each proposing different mechanisms for moving the source and/or detector, and aiming to reduce the acquisition time, total patient dose or cost of the technology. In the 6th and latest generation, the source emits radiations in a wide fan-shaped beam, while rotating up and around the patient in a helical motion [26].

Additionally, in the first generations of CT, only one row of detector was placed opposite to the source, allowing the acquisition of only one slice at a time. Since then, Multi-Detector CT (MDCT) and the more recent Cone-Beam CT (CBCT) made use of multiple rows of detectors that allowed the simultaneous acquisition of multiple slices. Both approaches have their strengths and limitations [32].

Using a powerful computing system to process the series of resulting 2D projections acquired, the original 3-dimensional function of the linear attenuation coefficient μ is then reconstructed. In mathematics, the process of calculating unknown causal factors from a set of observations is called an inverse problem. But such problems are ill-posed; a finite set of samples leads to an

infinite number of solutions. Nevertheless, both the quality and precision of the reconstruction grow with the number of projections used [33].

The result of the reconstruction process is a grayscale 3D image (called "volume") composed of many 3D pixels (called "voxels"). Each voxel bears a scalar value that represents its attenuation. When displaying the volume in a dedicated software, tissue with higher attenuation looks brighter, whereas tissue with lower attenuation appears darker [26].

The scalar value of a voxel is called the radiodensity, and is expressed in Hounsfield units [HU], a standard, dimensionless scale used universally in CT volumes. The formula to convert a given linear attenuation coefficient μ to Hounsfield units is [34];

$$\text{Value in HU} = \frac{\mu - \mu_{water}}{\mu_{water}} \cdot 1000, \quad (1.4)$$

where $\mu_{water} [cm^{-1}]$ designates the linear attenuation coefficient of distilled water at a temperature of 273.15 K (0°C) and a pressure of 10^5 Pa (1 bar).

The scale was designed such that the radiodensity of water is 0 HU, and that of air is -1000 HU [34]. Table 1.1 below summarizes common ranges of values, although the exact values sometimes slightly vary between CT systems manufacturers.

Table 1.1: *Typical values in Hounsfield Units (HU) observed when using CT systems. Emboldened lines represent reference values measured at 273.15 K and 1 bar. Adapted from [34].*

Matter	Typical value [HU]
Air	-1000
Lungs	from -950 to -650
Subcutaneous fat	from -115 to -100
Distilled water	0
Soft tissue (brain, muscles, ...)	from 0 to 100
Bones	from 300 (trabecula) to 2000 (cortical)
Metals	> 3000

Nowadays, most traditional CT systems encode the radiodensity of one voxel using 12 bits. This means that up to $2^{12} = 4096$ distinct values can be expressed. The standard practice is to measure radiodensity between -1024 and 3071 HU. Values outside this range (*e.g.* metals) are simply clipped to the bounds. More advanced systems sometimes use 16 bits to encode a voxel, allowing a window of up to $2^{16} = 65536$ distinct values. However, for such systems, the range of values covered by this window differs between manufacturers [34].

1.3 World and Image Coordinate Systems

As said above, the result of the reconstruction process is a 3D image of the radiodensity of the object of interest. Each voxel in the image represents a small volume in the real world. Figure 1.5 illustrates the principle of describing a given point using different coordinate systems, *e.g.* the CT's voxel space and real-world space. Hence, there needs to be methods to convert coordinates within the image to real-world coordinates, and *vice versa*.

To make the conversion from one orthogonal coordinate system (A) to another (B), the following transforms can be performed [35]:

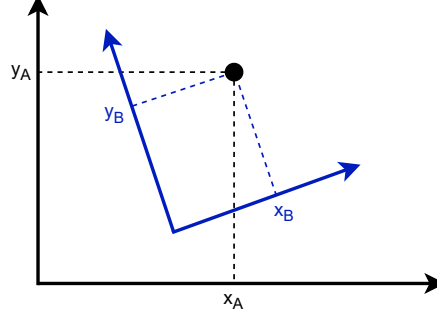


Figure 1.5: One point expressed in two different 2D coordinate spaces.

- **R**otate the axes of A and align them with B's;
- **ReScale** the basis vectors of A individually so that their norms match with B's;
- **T**ranslate A's origin to match with B's.

These three intermediary transforms can be expressed as linear operations through a series of matrices $\mathbf{R}, \mathbf{S}, \mathbf{T} \in \mathbb{R}^{4 \times 4}$. With these, converting coordinates from A to B can be expressed as

$$\mathbf{x}_B = \mathbf{T} \mathbf{S} \mathbf{R} \mathbf{x}_A, \quad (1.5)$$

where $\mathbf{R}$, $\mathbf{S}$, and $\mathbf{T}$ represent the three transforms listed above, and $\mathbf{x}_A$ (resp. $\mathbf{x}_B$) $\in \mathbb{R}^4$ represents the homogeneous coordinates of a point in the coordinate system A (resp. B). The concept of homogeneous coordinates is explained below. The developed form of the equation is [35]:

$$\Leftrightarrow \begin{bmatrix} x_B \\ y_B \\ z_B \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & t_x \\ 0 & 0 & 0 & t_y \\ 0 & 0 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}' & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ z_A \\ 1 \end{bmatrix}, \quad (1.6)$$

where the parameters t_i represent the translation parameters, and s_i the scaling parameters. $\mathbf{R}' \in \mathbb{R}^{3 \times 3}$ represents a standard 3D rotation matrix in $\mathbb{R}^3$. Applying these three transforms preserves parallel lines and orthogonality.

It is worth noting that the coordinates have four components here: the standard x, y, z, as well as an additional one at the end. These are called homogeneous coordinates, and they contrast with inhomogeneous coordinates, which only have the first three components. Using homogeneous coordinates to perform the conversion from one coordinate system to the other allows applying the translation transform using matrix multiplication instead of addition, just like rotation and scaling [35]. This allows the compact notation $\mathbf{x}_B = \mathbf{M} \mathbf{x}_a$, where the transform matrix $\mathbf{M} := \mathbf{T} \mathbf{S} \mathbf{R}$ summarizes all three intermediary transforms. The conversion from coordinate space B back to A is performed through $\mathbf{x}_A = \mathbf{M}^{-1} \mathbf{x}_B$, using the inverse matrix of $\mathbf{M}$.

When acquiring a CT volume, the transform matrix $\mathbf{M}$ that converts image coordinates to real-world coordinates is also added to the volume's file.

In section 1.2, two common modern technologies, CBCT and MDCT, were mentioned. One crucial difference between those is that the voxel size in CBCT volumes is isotropic (*i.e.* voxels are cubic), which is generally not the case in MDCT volumes (*i.e.* some sides of a voxel are longer than others). Therefore, an isotropic distance function defined in the MDCT image space

is not isotropic in the real world. In other words, two displacements over a given distance and in different directions in the image will generally lead to displacements over different distances in the real world. This can be an issue for distance-dependent problems. A workaround is preferring world coordinates to image coordinates when possible.

1.4 Relevant Artifacts in CT Scans

When dealing with CT volumes, several types of artifacts and imperfections can impact the quality and clarity of the image [26], [36]. This subsection briefly presents the main artifacts produced by heavy materials, such as metals. The severity of each artifact type can vary between CT volumes, and should preferably be handled during their preprocessing.

1.4.1 Beam Hardening and Scattering

Both of these artifacts share the same cause; in CT scans, the X-ray radiation is polychromatic, *i.e.* it contains photons with varying energies [26].

Since high-energy photons are less sensitive to attenuation (as previously shown in figure 1.2), they traverse material more easily. Therefore, even if a highly attenuating material is present between the source and detector, a small proportion of very high-energy photons are still able to traverse it and hit the detector, instead of being attenuated.

On another hand, low-energy photons coming from rays adjacent to the dense material are sensitive to Rayleigh and Compton scattering (as previously shown in figure 1.2). As a result, some of these photons are scattered, and their modified trajectories make them incorrectly hit the detector behind the dense object, instead of the on the photon's original path.

These two phenomenons result in a higher count of photons hitting the detected behind a dense object, making the material seem less attenuating than it really is. The resulting effect on a slice is the appearance of dark streaks in the directions of greater attenuation around the dense material, as well as darker values within the material itself. In addition, the backprojection algorithm used for reconstruction exaggerates differences between adjacent detectors, producing bright streaks in other directions [36].

Though modern scanners correct these phenomenons by calculating the average amount of beam hardening in the image and removing it, this is not enough to fully suppress beam hardening around high atomic number materials, such as metals.

1.4.2 Metal Artifacts

Metal artifacts are the aggregation of several interferences caused by two main causes: the metal itself and its edges. Since metallic materials have high atomic numbers Z compared to tissue, they have high linear attenuation coefficients, and hence produce by themselves beam hardening and scatter (discussed above), producing dark streaks between the metals and bright streaks in other directions. But additionally, at the edge of a metallic material, the sharp contrast between human tissue and the metal can cause other dark and bright streaks to appear all around the metal when running the reconstruction algorithm. This can be caused by either undersampling, patient motion, or inter-slice interpolation.

In the case where two electrodes have crossing trajectories, metal artifacts can also result in the apparent merging of contacts into a single mass of voxels where the intersection takes place [13]. Nowadays, there exist several techniques that attempt to reduce metal artifacts [37], each with

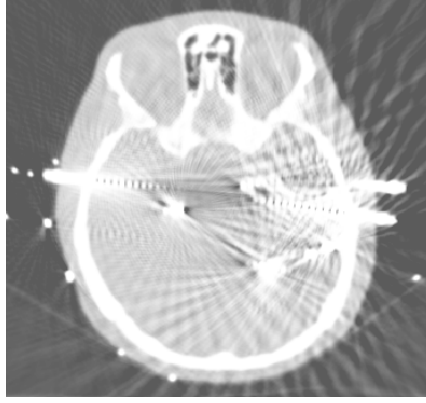


Figure 1.6: *A slice of CT scan with strong metal artifacts, observable as dark and bright stripes, present everywhere in the image, and with large blocks of bright pixels around the metallic materials.*

varying results and efficiency. Figure 1.6 displays an example of slice of CT scan with pronounced metal artifacts caused by intracranial depth electrodes, which impact the quality of the overall image.

Chapter 2

Stereoelectroencephalography

This chapter provides an overview of stereoelectroencephalography (sEEG), a critical technique used for monitoring local brain activity. It starts by briefly defining sEEG and describing depth electrodes. The chapter then explores the design and functionality of depth electrodes, as well as the general procedure that surrounds their implantation. Finally, it reviews existing tools and algorithms developed to accurately localize electrode contacts in the brain, highlighting both their strengths and limitations.

2.1 Basics of Stereoelectroencephalography

2.1.1 Principles of sEEG

sEEG describes a sub-category of invasive EEG that makes use of depth electrodes implanted in the patient's brain to record its electrical activity. It differs from standard electroencephalography (EEG), which is non-invasive and records brain activity from outside the patient's skull. It is also different from electrocorticography (ECoG), which uses so-called subdural electrodes, that consist of a grid of electrodes placed on the surface of the patient's cortex [38].

Depth electrodes are long, thin electrodes used to record the electrical activity of deep regions of the brain. They play a major role in monitoring seizure activity in patients with intractable epilepsy [39] and locating their epileptogenic zone (EZ). Figure 2.1 depicts how depth electrodes are used to record a patient's brain activity.

Figure 2.2 shows a depth electrode. It is a long, thin shaft that contains several measuring points, called contacts. While the shaft is inserted into the patient's head, its external end is attached to a long cable that leads to a series of connections, to retrieve the measured data using a computer.

2.1.2 Depth Electrodes

The shaft is composed of a biocompatible material that is semirigid or flexible, and usually has a diameter of 1 mm or less. During the electrode insertion procedure, the shaft is also fixated to the skull using a special type of screw, also called "bolt". Throughout the entire monitoring procedure, the shaft takes several responsibilities:

- It maintains the contacts in place within the patient's head. The brain sometimes exhibits natural micromotions, that cause the contacts to slightly deviate from their planned

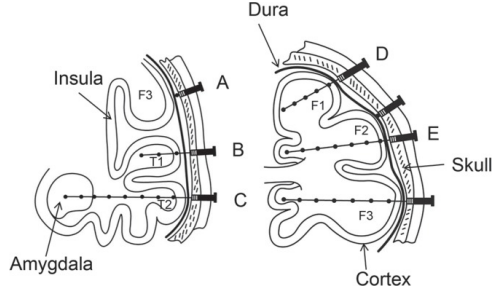


Figure 2.1: *Illustration of the general concept of stereoelectroencephalography. Depth electrodes (B, C, D, E) are implanted into a patient's brain to record subcortical activity [40].*

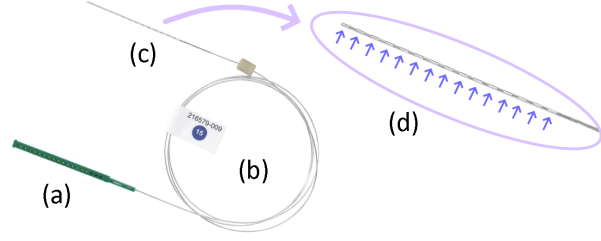


Figure 2.2: *Components of a depth electrode. (a) the connectors, that can be connected to the measuring device, (b) the cable, (c) the semi-rigid shaft implanted in the patient's brain, (d) the contacts in the shaft. Adapted from [41].*

position. The shaft keeps the displacement as small as possible and with minimal stress induced on the brain, to avoid injuries [42].

- It takes care of the electrical insulation between brain tissue and the microcables running through the electrodes, which transfer the measured activity to the connections outside the patient's skull.
- It eases the insertion and removal procedures of the electrodes. The former is performed using a rigid stylet, removed upon completion of the insertion [42].

Depth electrodes used for sEEG typically have between 4 and 18 contacts, all spaced equally, with distances ranging from 2 to 10 mm apart [40]. They are most often inserted as orthogonally as possible through a standard burr hole or a twist drill hole [40]. The number of electrodes and their placement is specific to each patient, and decided by the neurosurgeons. The electrodes are placed as to optimize a list of criteria such as minimizing the distance to the regions suspected of being the EZ, minimizing the angle of insertion of the electrodes into the skull, or avoiding the traversing of vessels and sulci [43].

The contacts are usually made of a platinum-iridium alloy, which provides a conveniently low impedance and eases the transmission of the measured signals through the electrodes from the contacts to the connections. Furthermore, since neither of these two metals is magnetic, they are not affected by strong magnetic fields. Hence, a patient with depth electrodes implanted can undergo a Magnetic Resonance Imaging (MRI) procedure without any risk of harm or injury; sEEG is MRI-compatible.

MRI is a non-invasive medical imaging technique, like CT. But contrary to CT scans, MRI does not use ionizing radiation. Instead, it uses strong magnetic fields to produce detailed images of the inside of the body. It works by aligning hydrogen atoms in the body with a magnetic field, then using radiofrequency pulses to disturb this alignment. Eventually, the atoms return to their original state. The time it takes for an atom to recover this state varies between tissues. These differences in the recovery times are then measured and processed by a computer to produce an image. MRI shows particularly good contrast between soft tissues like the brain, muscles, and organs [26].

Nowadays, intractable epilepsy can be treated through ablation of the EZ. But prior to this intervention, it is essential to precisely locate it. As discussed earlier, this is the objective that sEEG fulfills. The exact process prior to and during the implantation period of depth electrodes

can vary, but usually goes as follows:

1. An EEG or other non-invasive method is used to attempt to localize the EZ, but the results lack precision. The patient is thus prescribed an sEEG test instead.
2. Presurgical CT and/or MRI scans of the patient are acquired and used to study the anatomy of their brain.
3. The neurosurgeons use the information gained from the scan(s) and their hypotheses about the location of the EZ to design a plan for the placement of several electrodes. The plan details how many electrodes are placed, their entry points, their angle of insertion, and their lengths (*i.e.* number of contacts on each).
4. The electrodes are implanted into the patient's head following the plan during a surgery.
5. Post-surgical CT and/or MRI scans are acquired to view the electrodes in the patient's brain, in case of slight deviations from the plan.
6. The patient stays at the hospital for a few days, usually about a week, until one or several seizures occur. During the stay, the patient's brain activity is constantly recorded by the electrodes.
7. After the observation period ends, another surgery takes place to remove the electrodes from the patient.
8. Analysis of the patient's brain activity throughout the stay and around the seizures allow the neurosurgeons to locate the EZ.

For presurgical scans, MRI scans are preferable, as they provide better contrast for the soft tissue that composes the brain. However, CT scans can also be used for this task.

For post-surgical scans, both CT and MRI can be used to evaluate the precise position of the contacts in the patient's brain. An example of the former is illustrated in figure 2.3. Table 2.1 presents a non-exhaustive list of strengths and limitations of each modality in the context of electrode localization.

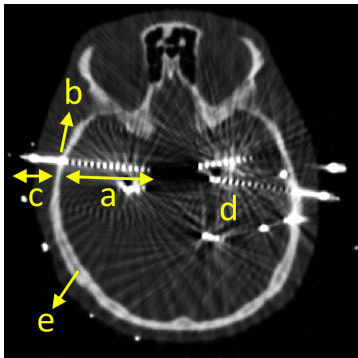


Figure 2.3: A slice of post-surgical CT scan, with sEEG-related concepts annotated: (a) electrode contacts, (b) entry point, (c) screw, (d) other electrodes, (e) skull.

Table 2.1: A comparison between the CT and MRI modalities for post-surgical evaluation of depth electrode implantation.

CT	
Pros	Compatible with all depth electrodes More commonly found in medical facilities Fast acquisition (a few minutes a most) Covered by most contact localization tools
Cons	Exposes the patient to radiations Metal artifacts can affect the image's quality
MRI	
Pros	Compatible with most depth electrodes Easier analysis of the contacts' anatomical positions
Cons	Not present in all medical facilities Slow acquisition (15-90 minutes) Artifacts can make it difficult to distinguish contacts

2.2 Locating Contacts and Existing Tools

When sEEG is used as a means to the treatment for a patient’s epilepsy, the EZ is found by analyzing the brain activity recorded by the contacts. But to effectively locate the EZ within the patient’s brain, it is essential to know the precise position of all contacts as well. To do that, after acquiring the post-surgical scan, the neuroscientists need to manually locate all 100 to 250 contacts along the implanted electrodes, accounting for the inter-contact distance, the angle of insertion of the electrode, potential shifts with respect to the pre-operative plan, and more. Processing a single image takes between one and two hours in total. In addition to being tedious and time-consuming for the neuroscientist, the process is also very error-prone. Furthermore, the complexity of the task increases when electrodes are oriented vertically or surrounded by strong artifacts at intersections [13]–[17], [19], [20].

To address these challenges, researchers have developed various automated and semi-automated methods for electrode contacts localization [13]–[21]. These methods aim to accelerate the workflow, improve reproducibility, and ease the clinicians’ task. All methods detailed here start with a thresholding of the CT image to segment the electrodes from the rest of the head, although the value of the threshold differs. Some papers also co-registered the post-surgical CT scans processed to the presurgical MRI scans, to allow identifying the structural regions of the brain in which the contacts are located [13]–[16], [18], [20], [21]. Among these, some papers [15], [18], [20] additionally co-registered the MRI scan to the Montreal Neurological Institute (MNI) space, a standard coordinate space for brain images used to create structural or anatomical atlases of the brain [12].

In 2015, Arnulfo et al. [13] proposed an automatic process that relied on the pre-operative plan to estimate the axes of the electrodes from their planned entry and target points. Following that trajectory, a geometrical-constrained search algorithm was used to estimate the exact position of the contacts, accounting for potential bending or crossing of electrodes. However, the study implicitly relied on the hypothesis of weak metal artifacts, where all segmented contacts in the image lie in their own connected component. However, in the presence of strong artifacts, some neighboring contacts can be joined within the same connected component, especially around intersections.

In the same year, Meesters et al. [14] located the screws on the patient’s skull using a multi-scale filter to automatically compute the axes of the electrodes. The contacts were then retrieved from connected components around the axes based on their geometry or intensity. Contrary to the other methods detailed here, which segmented the electrodes from the rest of the head with thresholds ranging between 1500 and 2600 HU, Meesters et al. [14] used a threshold of only 500 HU, which voluntarily let the skull into the processed binary image. Though the approach was fully automated, it also relied on the strong assumption of weak artifacts. The article also did not specify how intersections were handled by the algorithm. But since the patient’s suspected EZ is often targeted by several depth electrodes, intersections can be common.

In 2017, Qin et al. [15] proposed a semi-automatic method for both ECoG and sEEG, that performed clustering of the electrodes. The clusters were then manually corrected to exclude portions of the skull that could be unexpectedly included in the clusters. Then, the paper relied on a method similar to Arnulfo et al.’s [13] algorithm to retrieve the two ends of each electrodes. Though this method accounted for bent electrodes up to a cubic curve, the paper also relied on a clean separation between the contacts in the CT scan.

In 2018, Granados et al. [16] implemented a fully automated algorithm in C++ that did not rely on any prior information about the electrodes such as inter-contact spacing, entry points,

or number of contacts. The algorithm started by estimating the electrodes' axes from the screws, then performed a search constrained by angle and distance along these axes to locate the electrodes. A post-processing step took care of the unclassified contacts. A key feature of the algorithm was that it could function without any information about the electrodes. Similarly to the others, the method was robust to bending electrodes, but failed to consider joint contacts and intersections.

In the same year, Benadi et al. [17] proposed a fully automated method by also using the screws to compute the electrodes' major axes, then classifying all contacts by proximity to the axes. An additional feature was added to handle close electrodes, but the method still suffered from the same weaknesses as the solutions proposed by Meesters et al [14], Granados et al. [16], and Qin et al. [15].

In 2019, to patch the flaws of the preceding methods, Lin et al. [18] developed a pipeline with several stages of processing relying notably on mathematical morphology, a novel approach for the task of localizing electrodes. Essentially, the process began by grouping contacts within the same electrode, then identified crossing electrodes, assigned contacts to their respective electrodes, and finally distinguished contacts located on the same electrode. Out of all the methods cited here, this was the first one to propose a solution robust to CT images where several contacts remain joint after the preliminary thresholding. Furthermore, it explicitly handled the case of crossing electrodes. However, the method only considered mild, negligible electrode bending. Furthermore, it required to know the inter-contact distance beforehand, *i.e.* the specifications of the electrodes used in the image.

In 2022, Pantovic et al. [19] leveraged the advances in artificial intelligence, and particularly deep learning in image processing, to propose two U-Net models [44], [45] to segment the contacts within a CT volume and return a binary mask of these. One model was trained to find all contact pixels within a single 2D slice, whereas the other was trained to segment contact voxels in entire 3D volumes. Like the method proposed by Granados et al. [16], the models did not require any prior information about the electrodes for the prediction of the contact masks. The results showed that the models complemented each other's weaknesses: the 2D model was weak to strong artifacts that the 3D model could ignore, and the 3D model showed difficulty to segment the most outer contacts in the brain, contrary to the 2D model. However, the training of deep learning architectures such as U-Net requires sufficient images and the specifications of the electrodes used. Furthermore, a plain U-Net model can only output a mask of the contacts, and an additional step is thus required to extract their exact coordinates. Finally, the models were trained and validated using the same dataset. As a result, the reported results may not reflect the actual performances of the solutions proposed in general cases.

In 2024, Wong et al. [20] proposed a novel approach for the localizing and labeling of electrode contacts. After applying the usual threshold and a brain mask to the image, the algorithm drew lines between all pairs of connected components. Then, the line with the most contacts encountered along its path, while maintaining a consistent inter-contact distance, was selected and labeled as an electrode trajectory. The process was then repeated until all electrodes were found. For better precision in case of strong CT artifacts, an optional post-processing step was proposed, requiring the number of contacts along each electrode as well as the inter-contact distance. The approach was simple and intuitive, but at the cost of several assumptions and weaknesses. Firstly, representing electrodes as straight lines may not accurately capture those with moderate to significant bending. Secondly, the paper does not specify any action performed to classify correctly contacts seemingly merged around electrode intersections.

Nowadays, several applications dedicated to the processing of medical images also propose tools to accelerate the manual localization of depth electrodes. For example, Brainstorm, developed by Chinara et al. [21], allows the user to interact with the CT image in a 3D viewer, select manually the end points of the electrodes, and automatically add contacts on a line in between, based on the user-defined number of contacts, inter-contact spacing, contact width,... However, Brainstorm does not support curved electrodes at the time being, and those must be handled manually by the user.

Table 2.2 summarizes the key features of each algorithm presented in this section. It is worth noting that the table does not reflect the actual performances of the methods proposed, only the inputs they need and particular cases in sEEG that they can handle. Overall, it is difficult to design a fully automated algorithm that requires minimal prior knowledge about the image and/or electrodes while also accounting for all specific cases such as electrode bending, intersections, and strong CT artifacts.

Table 2.2: *A summary of the key features of the existing tools and methods studied. Individual performances are not reflected.*

Authors	Auto	NC	EP	ICD	BM	It	Ar	Bd
Arnulfo et al. [13]	F		●			●		●
Meesters et al. [14]	F	●			●			●
Qin et al. [15]	s			●	●			●
Granados et al. [16]	F							●
Benadi et al. [17]	F			●				
Lin et al. [18]	F			●	●	●	●	
Pantovic et al. [19]	F	*	*	*				●
Wong et al. [20]	F	●	●	●				
Chinara et al. [21]	s	●	●	●		●	●	

Acronym	Definition
Auto	Fully (F) or semi-automated (s)
NC	Requires number of contacts per electrode
EP	Requires coordinates of electrode entry points
ICD	Requires inter-contact distance
BM	Requires brain mask
It	Robust to electrode crossings
Ar	Robust to strong CT artifacts and/or joint contacts after thresholding
Bd	Robust to electrode bending
*	During training only

Chapter 3

Mathematical Morphology

This chapter briefly presents Mathematical Morphology (MM) and the basic operators that are later used in this report. Its content is strongly inspired by [3], [4]. It starts by defining MM and the concept of binary image. Then, it presents basic MM operators such as dilation, erosion, and reconstruction by dilation. Finally, it describes the process of ultimate erosion.

This chapter is aimed at readers with little to no knowledge about image representation and mathematical morphology. It features both formal equations and explanatory figures, as to ease the understanding of the concepts covered.

3.1 Basics of Mathematical Morphology and Binary Images

MM is a branch of image processing which aims to analyze the patterns and shapes observable in an image in order to extract useful information. It consists in a set of operators that can be applied directly onto the original image. This process contrasts with many other processing techniques, which first express the image into a new basis, and only then apply filters onto these new representations. For example, the Discrete Fourier Transform and Wavelet Transform use this operational mode. MM operators differ from these by processing the image in its native form: the spatial domain.

A binary image is defined as a function $f : D \subset \mathbb{Z}^n \rightarrow \{0, 1\}$ that maps each spatial position in D to a single bit. A binary image can also be represented as a set $I := \{x \in D | f(x) = 1\}$ that contains all the positions with a value set to 1. This set is also called the foreground or mask, and its complementary is called the background. In practice, the set of positions D is discrete and bounded, conveniently allowing a computer to store the sequence of bits that compose the image. The positions are typically called "pixels" in two dimensions, and "voxels" in three dimensions. To display the image, the foreground is typically represented in black, and the background in white. Figure 3.1 illustrates an example of binary image.

The general goal of MM operators is to extract useful information from an image. The choice of the operator depends on the task or the specific objectives: reducing noise, finding the occurrences of a specific pattern in an image, or more. Though MM operators can be defined for all types of images among binary, grayscale, and colored [47], this thesis will focus on the former.

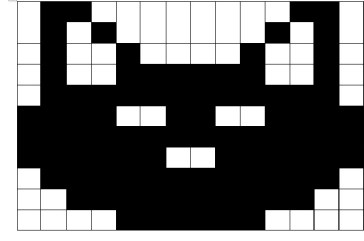


Figure 3.1: An example of binary image. The grid is only present for display purposes [46].

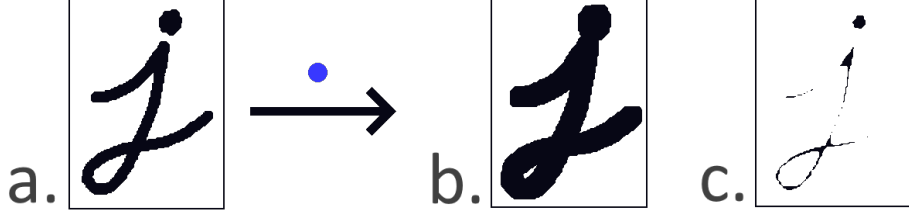


Figure 3.2: From an original image (a), an example of dilation (b) and erosion (c). The blue dot is the structuring element used for both. Adapted from [48].

3.2 Dilation and Erosion

Dilation and erosion are the two most fundamental MM operators, represented by the Minkowski sum $\oplus$ and subtraction $\ominus$ respectively. For binary images, they are defined as:

$$I \oplus B := \bigcup_{b \in B} I_b = \{x + b | x \in I, b \in B\} \quad (3.1)$$

$$I \ominus B := \bigcap_{b \in B} I_{-b} = \{p \in D | B_p \subseteq I\} \quad (3.2)$$

where $B \subset \mathbb{Z}^n$ is another binary image, *i.e.* set, called the structuring element and applied to I . In practice, the foreground of B is often smaller than that of I . The notation $I_b := \{x + b | x \in I\}$ denotes the translation of image I by $b \in B$.

If B comprises the origin of $D \in \mathbb{Z}^n$, then dilation has the effect of inflating the foreground with all positions that B , translated to anywhere in I , can overlap with. On the other hand, erosion results in reducing the foreground by keeping only the positions where the transposed structuring element is fully contained within I . Figure 3.2 shows an example of the effects of dilation and erosion on an image.

When applying successive dilations to an image, the foreground keeps growing with each operation, until it covers the entire image. On the contrary, successively eroding the image shrinks the foreground with each operation, until it becomes empty. Dilation and erosion are not reciprocal operators.

3.3 Image Reconstruction by Dilation

Image reconstruction [49] is another basic operation in MM. Reconstruction can be performed with either dilation or erosion, each producing different effects for different goals. In both cases, reconstruction is performed by applying successively the chosen operator, starting from a seed

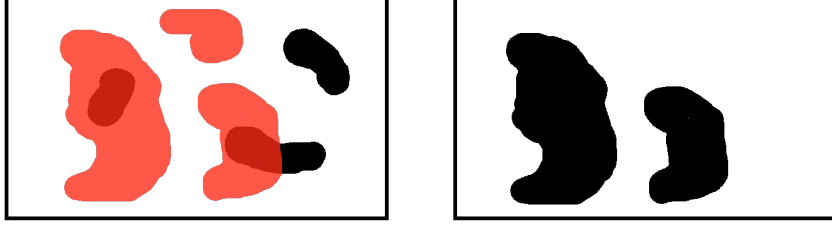


Figure 3.3: On the left, two overlapped images: a seed (black) and a mask (transparent, red). On the right, the result of image reconstruction with these inputs. Components in the mask that partly overlap with the seed get fully filled, and the others are ignored.

$R \subset D$ (also called marker) until a mask $F \subset D$ is reached. Both R and F represent binary images.

Reconstruction through dilation with seed R and mask F is denoted as $rec_{\oplus}(R, F)$ and is computed as in algorithm 1, where R_n denotes the result of n successive dilations of R within the bounds of the mask F .

Algorithm 1 Ultimate Erosion

```

function RECONSTRUCTION( $R, F$ )
   $R_0 \leftarrow R \cap F$ 
   $R_1 \leftarrow (R_0 \oplus B) \cap F$ 
   $n \leftarrow 1$ 
  while  $R_n \neq R_{n-1}$  do
     $R_{n+1} \leftarrow (R_n \oplus B) \cap F$  ▷ Dilate  $R_n$  within bounds of  $F$ 
     $n \leftarrow n + 1$ 
  end while
   $rec_{\oplus}(R, F) \leftarrow R_n$ 
  return  $rec_{\oplus}(R, F)$ 
end function

```

In this algorithm, $B \subset D$ can be any structuring element that comprises the origin. Usually, the area of foreground covered by R is smaller than that of F , and both consist in several connected components (CCs), some of them intersecting at least partially. The image resulting from reconstruction through dilation is the union of all CCs in F that intersect at least partially with the seed R , as illustrated by figure 3.3. If all pixels from a CC in F are missing from R , then the entire CC is missing in the output as well. Then, the seed R can be seen as the starting image, and F as an objective that reconstruction attempts to reach from R .

It is worth noting that the result of reconstruction is independent on the structuring element used, provided that it contains the origin. Since structuring elements are user-defined, this is never a problem in practice.

3.4 Ultimate Erosion

Ultimate erosion is a more advanced morphological operator that finds the seed of all the objects in an image I using a structuring element B . Here, the meaning of seed differs from the one defined in section 3.3, which was the starting point of reconstruction. In the context of ultimate erosion, seeds are the output. A seed designates the set of all irreducible pixels at the center of an object in I . Irreducible means that eroding the seed with B would make it completely disappear. In ultimate erosion, two different connected components are always considered as

different objects. But a single connected component can be considered as one or several objects, distinguishable by significant concavities in the edges of the image. Figure 3.4 illustrates an example of different objects contained in a single connected component, that ultimate erosion aims to separate.



Figure 3.4: *On the left, a binary image that consists of one connected component. On the right, the different objects in that image, distinguishable due to the concavities in the foreground's edges.*

Ultimate erosion is performed by successively eroding the image I . When a connected component gets fully eroded, its last state before total disappearance (*i.e.* its seed) is saved to the output. Algorithm 2 defines the recursive implementation of ultimate erosion.

Algorithm 2 Ultimate Erosion

```

function ULTIMATE_EROSION( $I$ )
   $I_0 \leftarrow I$ 
   $n \leftarrow 0$ 
  while  $I_n \neq \emptyset$  do
     $I_{n+1} \leftarrow I_n \ominus B$                                  $\triangleright$  Erode  $I$  slightly
     $R_n \leftarrow \text{rec}_{\oplus}(I_{n+1}, I_n)$                      $\triangleright$  The image  $I_n$  without the components that got fully eroded
     $D_n \leftarrow I_n \setminus R_n$                              $\triangleright$  The components that got fully eroded at iteration  $n$ 
     $n \leftarrow n + 1$ 
  end while
  return  $\bigcup_{n \in \mathbb{N}} D_n$ 
end function

```

In the algorithm, I_n represents the image after n successive erosions. R_n is the reconstruction of I_n from I_{n+1} . It has the same content as I_n , except for the CCs that didn't survive the $(n+1)$ -th erosion, which are stored in D_n . One can observe the properties $R_n \subseteq I_n$ and $I_{n+1} \subset I_n$ while $I_n \neq \emptyset$, at which point the recursion is stopped. The end result is the set of all connected components right before their final erosion. Figure 3.5 shows a detailed example of the algorithm applied.

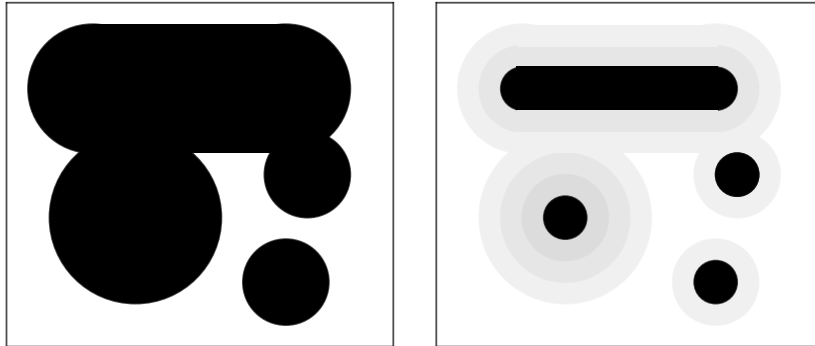


Figure 3.5: *Example of the ultimate erosion operator. The input is on the left, and the output on the right. Objects get separated into irreducible components at their centers. The gray areas display the successive erosions applied to the foreground. They are not part of the output.*

In 2D (resp. 3D), the structuring element B is usually a 3x3 square (resp. 3x3x3 cube) or a 3x3 cross (resp. 3x3x3 3D cross), to erode the edges of the image's foreground one pixel at a time. The difference between the two is the former also considers diagonals during the erosion. Usually, bigger structuring elements with respect to the size of the initial image lead to bigger seeds and a smaller number of them, although this is not always true.

It is worth noting that, while ultimate erosion is a very easy technique for computing the seeds of the objects in an image, it also suffers from some weaknesses:

- In small initial images, the presence of noise on the edges can lead to different results, notably the production of more seeds than there should be.
- Ultimate erosion is only applicable to foregrounds with no holes in them. Otherwise, they get eroded from inside as well and the algorithm fails to find a reasonable seed.
- The choice of structuring element can impact the shape, size and number of seeds produced in the output, especially in small images. Therefore, it must be chosen carefully.
- Without optimization, the algorithm's complexity is $\mathcal{O}(L^2W^2H^2)$ for a 3D image of length L , width W and height H . Fortunately, optimized variants exist to reduce the complexity in practice. But ultimate erosion can still be slow for big images that are mainly covered by foreground.

These drawbacks can be avoided with appropriate preprocessing, such as morphological closing, *i.e.* dilation immediately followed by erosion with the same structuring element, which removes small holes inside the foreground and small concavities on its edges.

Overall, ultimate erosion is a powerful tool for quickly computing the center regions of the objects in an image. It is particularly efficient with objects in distinct connected components and with concavities in their boundary. Sometimes, it can lead to false positives, where one single object incorrectly produces different seeds.

Chapter 4

Graph Theory and Search Algorithms

This short chapter defines the fundamental concepts of graph theory used in this thesis. Many major algorithms in use nowadays appeal to the concepts of graphs and searching over graphs. As this thesis makes use of graph theory, this chapter is aimed at readers who do not know or know little about it. It starts by defining graphs and several associated concepts. Then, it presents a framework for performing search on a graph in order to systematically retrieve optimal values.

4.1 Basics of Graph Theory

The first notions of graph theory date back to 1736, with the work of Leonhard Euler [50]. Since then, the subject has been widely studied and documented. Although the content of this section is the aggregation of multiple resources, it was mainly inspired by [5], [6].

4.1.1 Defining a Graph

In many real-world situations, relationships between objects can be viewed as a set of points connected by lines. For example, points could represent the inhabitants of a city, and the lines joining two points would represent that two inhabitants know each other. Such points and lines are respectively called vertices (or nodes) and edges. A graph $G := (V, E)$ is formally defined as a set of vertices V and a set of edges E . Vertices can represent any abstract object or concept, and edges are pairs of vertices $\{\{v_1, v_2\} | v_1, v_2 \in V\}$.

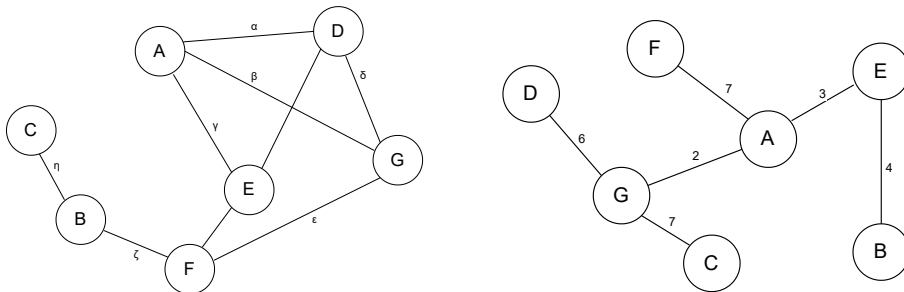


Figure 4.1: *Two examples of graphs: unweighted on the left, and weighted on the right. The Latin letters represent the vertices, and the Greek letters the edge names. Including the latter is optional.*

A weighted graph $G := (V, E, w)$ is a particular type of graph in which all edges are associated to a scalar value using a weighting function $w : E \rightarrow \mathbb{R}$. For example, a weighted graph could model all airports across the world (the vertices), whether or not these airports have connecting flights (the edges), and the distance of these flights (the weights). Figure 4.1 illustrates two examples of graphs, one unweighted and one weighted. When displaying a graph, the positioning of the nodes is arbitrary.

Here some additional definitions relative to graph theory:

- Trail: non-null sequence $P = (v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k)$ that alternates between vertices $v_i \in V$ and edges $e_i \in E$ such that e_i and e_{i+1} are both touching v_i , and such that all edges $\{e_0, \dots, e_k\}$ are distinct.
- Cycle: specific case of trail in which $v_0 = v_k$. In other words, a cycle is a trail that, if followed, leads back to the origin of the trail without ever traversing the same edge twice. In the graph on the left of figure 4.1, the trail $(A, \alpha, D, \delta, G, \beta, A)$ is a cycle.
- Connected graph: a graph is said to be connected if, for any given pair of vertices $v_i, v_j \in V$, there exists a trail such that $v_0 = v_i$ and $v_k = v_j$.

4.1.2 Trees and Minimum Spanning Trees

In graph theory, a tree is defined as a connected graph that contains no cycle. The graph on the right of figure 4.1 is a tree. A tree T can be obtained from a connected graph G by removing certain edges from E such that no cycle remains. Therefore, if the tree $T = (V, E')$ is obtained from the connected graph $G = (V, E)$ with this method, then $E' \subseteq E$. In most cases, there exist several trees that can be obtained from a given graph.

In weighted graphs, the minimum spanning tree of a connected graph $G = (V, E)$ is the tree $T = (V, E')$ obtained from that graph that has the lowest sum of remaining weights. In other words, $E' := \arg \min_{E'' \subseteq E} \sum_{e \in E''} |w(e)|$ such that (V, E'') is a tree.

Kruskal's algorithm is an efficient way for computing the minimum spanning tree of a graph. Its most basic implementation is defined in algorithm 3. It starts by sorting the edges by increasing weight and start with an empty set of selected edges. Then it iterates over the sorted edges, and adds edges to the selected set if they don't create a cycle with the other edges in the set. After the iterations end, the selected edges are those that constitute the minimum spanning tree of the graph [51].

Algorithm 3 Kruskal's Algorithm

```

function KRUSKAL(vertices, edges, weights)
  sorted_edges  $\leftarrow$  SORT_INCREASING(edges, weights)
  tree_edges  $\leftarrow \emptyset$ 
  for all  $e \in$  sorted_edges do
    if HAS_CYCLE(vertices, tree_edges  $\cup \{e\}$ ) then
      pass
    else
      tree_edges  $\leftarrow$  tree_edges  $\cup \{e\}$ 
    end if
  end for
  return tree_edges
end function

```

4.2 Graph Searching with Breadth-First Search

In graphs, vertices (nodes) can represent anything. In various situations, nodes characterize concepts that can be assessed by a scalar value, then compared. For example, if the nodes in a graph represent airports, then the value of each could be defined as the number of daily flights departing from that airport, the yearly number of passengers, or the total surface area occupied by its infrastructure.

Graph search, in its most general sense, is the mechanism used to systematically visit all the nodes in a graph [7]. It is often used as a mean to find the most optimal node in the graph, based on a pre-defined metric. One algorithm to perform graph search is Breadth-First Search (BFS). It works by starting at a node, defined as the root, then visiting all of its neighbors. Then, starting from these nodes, all their unvisited neighbors are visited as well. This logic is repeated until all nodes in the graph have been visited [52]. This procedure corresponds to traversing the graph in successive layers, beginning with the root node, followed by its immediate neighbors (first-order), then their neighbors (second-order), and so on, continuing iteratively until all reachable nodes have been visited, as illustrated in figure 4.2. Sometimes, graphs can also grow large in size, and only a relevant subset of neighbors are then considered for visiting, *i.e.* only part of the edges in the graph can be traversed during the search. The subset of neighbors that can be visited from a given node are said to be its children.

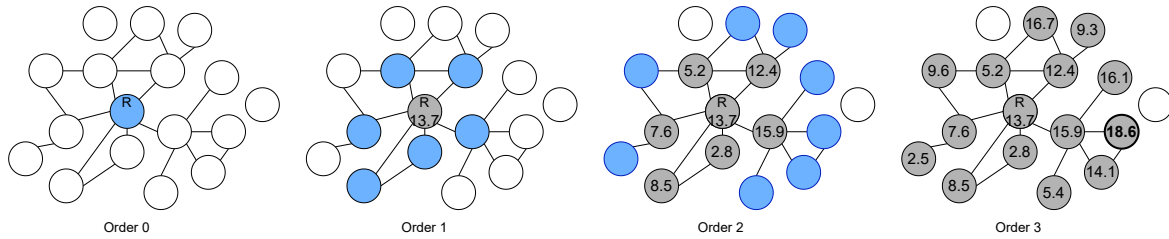


Figure 4.2: Example of the Breadth-First Search principle applied to a graph, from root node R , to retrieve the highest-scoring node. One layer (the blue nodes) is visited at a time. Gray nodes have been visited in the past.

Each visit consists in two actions: evaluation and expansion. Evaluation is the process of applying a metric function to the visited node to uncover its value. This step is used to keep track of the best node found yet during the search. Expansion is the action of retrieving the list of all the node's children.

Overall, to apply the BFS algorithm to a graph and retrieve an optimal node on a restricted subset of the graph (for efficiency reasons), the following concepts must be defined:

- **Root:** the node from where to start the search;
- **Children(n):** a function that, given a node n , returns the neighbors that can be reached and visited from n ;
- **Metric(n):** the evaluation function used to assess the scalar value of a node n ;
- **Is_Target(n)** (optional): a function that, given a node n , returns whether that node can be considered optimal or not, independently of its score. The optimal score returned by BFS is then the best score found among the target nodes.

After defining these concepts, running the BFS algorithm on a graph returns its best-scoring node.

Part II

Pipeline for Semi-Automated Detection of Electrode Contacts

Chapter 5

Materials

This chapter presents the dataset of CT images used for the development and validation of the algorithm presented in chapter 6. It also explains how the dataset was preprocessed to fit the input requirements of the algorithm.

5.1 Data Acquisition

For the development and validation of the pipeline presented in this paper, the CT scans from two patients, one male and one female, were used. The patients were both diagnosed with drug-resistant epilepsy and were undergoing a pre-surgical evaluation at the Rigshospitalet, Copenhagen, through the implantation of depth electrodes. The aim of their visits was to localize their EZ, so that a later surgery could ablate them. Table 5.1 details the information of the post-implantation CT images acquired. However, the model of the electrodes used was not available, restricting useful information such as the material of the contacts or the inter-contact spacing. Both CT images use the standard 12-bit quantization for encoding the voxel values.

Table 5.1: *The resolution in millimeters and total size in pixels of the CT images in the dataset.*

Patient	In-slice resolution	Slice thickness	Size
1	0.61 mm	1 mm	512 x 512 x 161
2	0.47 mm	0.625 mm	512 x 512 x 257

Patient 1 was implanted with 13 electrodes, each containing from 9 to 16 contacts. Patient 2 had 8 electrodes with a number of contacts ranging from 12 to 17.

The objective of the algorithm developed is to retrieve the position of all contacts in a CT image, classify them based on the electrode they belong to, and order them by depth along each electrode. Hence, for each patient, a ground truth was made by manually analyzing the images and retrieving the following pieces of information for all contacts:

- the coordinates—of the corner with lowest coordinates—of the center voxel;
- its electrode identifier (EID), an integer that uniquely identifies an electrode;
- its positional identifier (PID), an integer that uniquely identifies contacts within electrodes.

The PIDs within an electrode are sorted by depth, such that the deepest contact of the electrode is given the PID 0, the second-deepest contact gets PID 1, and so on. The complete data for each ground truth was stored in a Comma-Separated Values (CSV) file.

5.2 Preprocessing

To make the content of the dataset suitable for the pipeline presented in this thesis, some preprocessing had to be performed. It consisted in two steps, detailed in the following subsections.

5.2.1 File format

The initial dataset was provided in the Digital Imaging and Communication in Medicine [53] (DICOM) file format. DICOM is the single international standard for exchanging and storing medical images. Files stored with this standard conveniently store information about both the content of the image, and the metadata regarding its acquisition. For example, the latter consists in the type of image (CT, MRI, ...), the model of the device used, the affine matrix (see section 1.3), and more. In DICOM, all the slices of a CT image are stored into separate files.

The pipeline presented later in this thesis is fully written in the programming language Python, chosen for its simplicity and the many mathematical libraries it offers. Some libraries like `pydicom` [54] ease the reading and processing of DICOM files, but only allow loading one slice at a time, instead of the entire volume at once. Thus, loading an entire CT image with the DICOM format is impractical.

To fix this inconvenience, as a first step of preprocessing, the CT images of the dataset were converted from DICOM format (`.dcm`) to the "Neuroimaging Informatics Technology Initiative 2" [55] format (NIfTI-2, or simply NIfTI, `.nii`), which is another major file format used in neuroimaging to store medical images. Similarly to DICOM instances, a NIfTI file stores both the image itself and some metadata about the acquisition, notably the affine matrix. One main advantage of the NIfTI format over DICOM is that it stores a CT volume entirely into a single file, making it easy to read and process using dedicated Python libraries such as `nibabel` [56].

The conversion of the CT images from DICOM format to NIfTI-2 was performed using `dcm2niix v1.0.20220720` [57].

5.2.2 Brain Mask

Head CT volumes capture the entire region around the patient's skull. In the context of implanted depth electrodes, this means that both the intracranial shaft and the external wires are contained in the image, as shown by figure 5.1. Since the aim of this thesis is to retrieve the position of the electrode contacts within the patient's brain, only the former constitutes interesting information.

Therefore, the second preprocessing step is to compute a mask of the patient's brain, to exclude all regions of the CT where electrode contacts cannot be found. This was done using FreeSurfer's feature SynthStrip [58], a tool for removing non-brain areas from various modalities of head medical images. The tool also supports CT volumes with implanted electrodes. This skull-stripping process allows eliminating the electrodes' cables and fixation screws, useless for the task at hand. This step eases the processing of the image by the pipeline presented in this thesis, and reduces the size of the data to process.

Figure 5.2 illustrates the slice of a CT scan and the brain mask in that slice. Here, the mask computed presents imperfections, where skull voxels are unexpectedly included and a small region

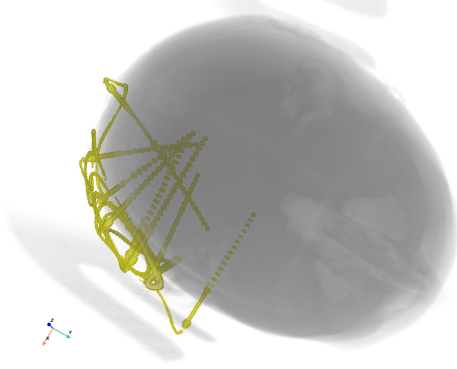


Figure 5.1: *The top view of Patient 1's transparent, 3D CT image. The opaque, yellow voxels are those with radiodensity higher than 2500 HU. They are composed of relevant intracranial electrodes, and cables and screws that must be ignored.*

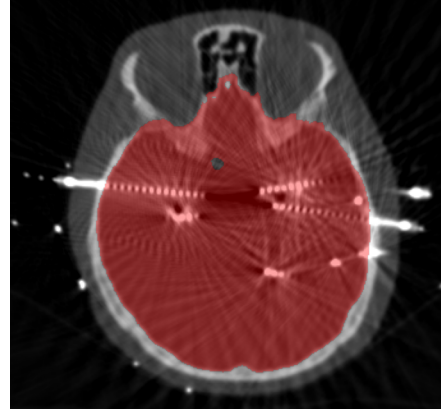


Figure 5.2: *A slice of Patient 1's CT scan (gray) and the brain mask computed by SynthStrip [58] (red). Some imperfections can be seen, where the mask wrongly includes skull voxels.*

of the brain is excluded. However, in the context of contact localization, a perfect brain mask is not required for the algorithm to function. What is of utmost importance is that all intracranial contacts are included in the brain mask, and that all external components (screws and cables) are excluded. Since figure 5.2 shows that the brain mask respects these two requirements, it is satisfactory for this use case.

Chapter 6

Methods

This chapter presents the pipeline of the algorithm developed for this thesis. First, the three major modules that constitute it are introduced in detail, each in a dedicated section. Each section ends with a summary of the module's main stages. Afterwards, a summary of the algorithm from a global point of view is provided. Finally, the chapter ends by defining some procedures and metrics used to assess the performance of the algorithm.

6.1 Extraction of Meaningful Points

The ultimate goal of the global pipeline is to precisely locate the contacts in a CT scan and tie them to their respective electrode. To do that, the pipeline regresses straight or curved line models to represent the electrodes, then computes optimal positions along those lines to place the contacts. However, regression tasks require a set of point coordinates as input. The CT volumes do not hold such information; they store scalar values in a 3-dimensional grid of voxels. Therefore, the first task of the pipeline is to transform the voxels of the input CT scans into a set of points with meaningful information about the electrodes. This section details the procedure followed to compute these meaningful points, which are defined as "centroids".

6.1.1 Segmentation of the Electrodes in the CT Image

Before further processing of the electrodes and contacts within the image, it is necessary to separate the electrode voxels from the voxels of the rest of the image (air, brain tissue, cerebrospinal fluid, ...). This task is called the segmentation of the electrodes. To achieve that, the properties of CT scans presented in section 1.2 are exploited.

In CT scans, voxel values represent radiodensity and are expressed in HU. As shown in table 1.1, all types of human tissue found in the head range from approximately -100 to 2000 HU. These values may slightly vary between different CT images, depending on the machine used to acquire them. On the other hand, electrodes contacts are metallic, and produce voxel values higher than 3000 HU. There is thus a wide gap of radiodensity between the patient's head and the electrodes to segment. The histograms in figure 6.1 showcase this gap before and after masking the intracranial region.

Thus, the simplest yet efficient way segment the electrodes from the rest of the head consists in classifying the voxels into the classes "electrode" or "head" based on their radiodensity; values higher than a threshold are considered electrode voxels, and the others are considered head

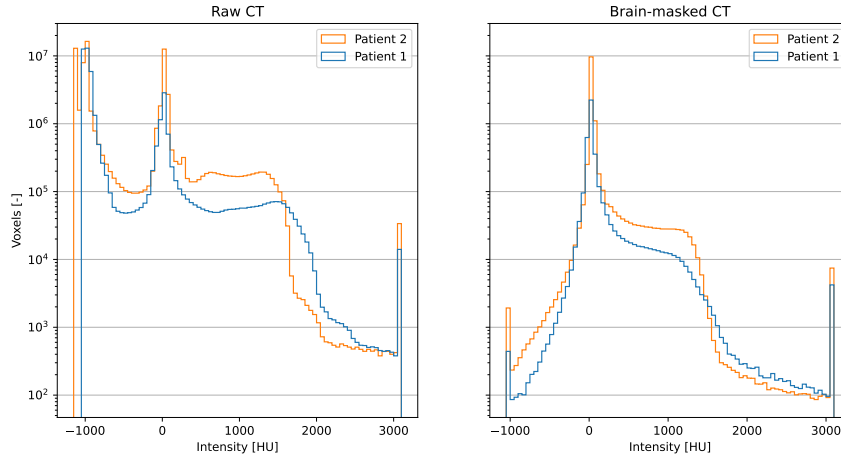


Figure 6.1: *The logarithmic histograms of both patients' CTs, before and after applying the brain mask. The histograms drop significantly between 2200 (skull) and 2900 (electrodes), as suggested in table 1.1.*

voxels. This method is called hard thresholding. It is the most basic procedure for segmenting an image, and is often too simple for most real-world, complex image processing tasks. However, in this context, the wide gap of intensities between the electrodes and the head in figure 6.1 justifies the adequacy of hard thresholding for the segmentation of electrodes in a CT scan.

To determine a satisfactory value for the threshold, some criteria must be taken into account:

- The threshold must not allow any skull voxel into the foreground, in case the frontier of the brain mask is not perfect and mistakenly contains some.
- The threshold must be robust to metal artifacts.
- The threshold must be low enough to allow a sufficient amount of electrode voxels into the foreground, as to not lose information about the contacts. Since most electrode voxels are clipped to the upper bound value of the CT image, as explained in section 1.2, this criterion is less restrictive than the other two.

The image on the right of figure 6.2 shows a slice of Patient 1's CT quantized to non-uniform ranges of radiodensity and illustrates the content of table 1.1. The image on the left is the initial slice. It can be observed that no voxel of the skull ever takes a value greater than 2400 HU, unless adjacent to an electrode. Additionally, all the contact centers are always greater than 2900 HU. Some centers may appear very small, like the contacts on the top left electrode, near the entry point. This is only due to the angle of insertion of the electrode, which is not perfectly parallel to the slice. Hence, those voxels only represent the top of these contacts. Finally, the figure shows that metal artifacts are, in fact, always composed of low radiodensities, below 1200 HU. These observations remain true across all the slices of the volume.

Supported by figures 6.1 and 6.2, the hard threshold applied to the image was fixed to 2500, which is in accordance with table 1.1 and provides a sufficient safety margin with respect to the skull. The threshold is applied onto the 3-dimensional brain mask, such that the foreground of the resulting binary image represents a mask of the electrodes inside the patient's brain, as shown in figure 6.3. Later in this report, it will be referred to as electrode mask (EMSK).

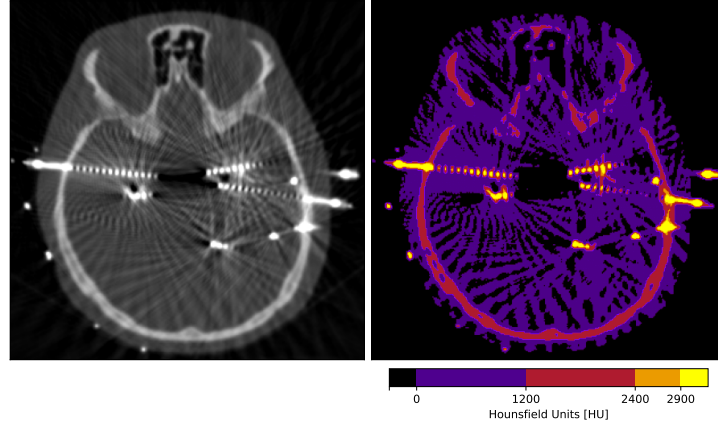


Figure 6.2: *On the left, an arbitrary slice from Patient 1's unprocessed CT volume before the brain mask. On the right, the quantized version of this slice, with boundaries carefully selected to showcase regions of the head and their values. The bright points outside the patient's head come from the cables of other electrodes. Values between 2400 and 2900 HU are well-suited for segmenting the electrodes from the rest of the head.*

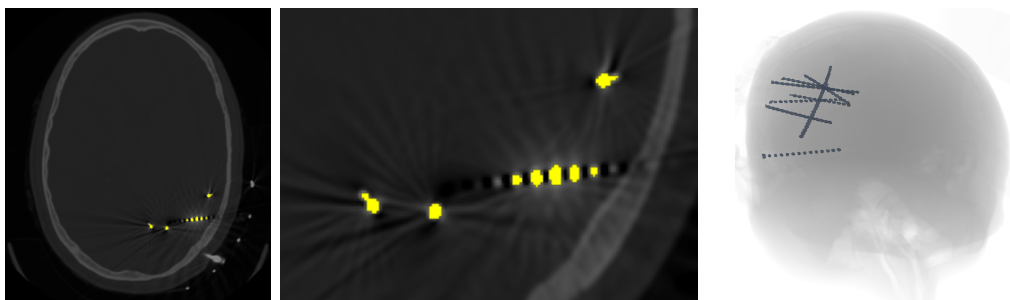


Figure 6.3: *(left) the yellow electrode mask over a slice of Patient 2's CT scan, darkened for display purposes; (center) a zoom of the first image on the electrodes; (right) a 3D view of the entire electrode mask (opaque, blue) within the CT scan (transparent, gray).*

It is worth noting that, at this point, the binary EMSK contains several connected components (CC). The nature of their content can vary; some CCs may contain exactly one contact, whereas others might contain entire electrodes or even groups of intersecting electrodes. Figure 6.4 represents these CCs with different colors, to emphasize the content of each.

Whether two neighboring contacts get separated into two different CCs or not depends on the radiodensity measured in the voxels in between (*i.e.* the shaft voxel) and the severity of the CT's metal artifacts. For example, in electrodes inserted at a small angle ($< 5^\circ$) with respect to the X-Y plane of the image, beam hardening artifacts cause the shaft voxels in between to appear as dark. Such regions are then cut out by the application of hard thresholding, resulting in neighboring contacts lying in different CCs. However, for electrodes inserted at an angle with respect to the X-Y plane (*e.g.* 45°), some shaft voxels do not lie on the same slice as their bordering contacts, and therefore do not suffer from the in-slice beam hardening artifacts produced by those. As a results, the values of the shaft voxels represent their actual radiodensities, which allows them to pass the threshold, connecting the two bordering contacts and making them form a single CC.

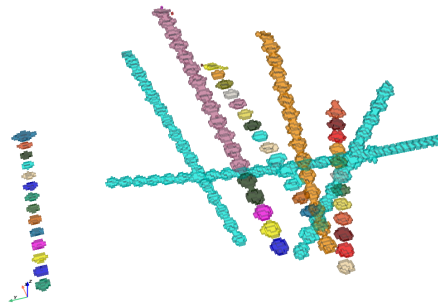


Figure 6.4: *The same electrode mask as presented in figure 6.3, from a different angle. Each connected component is mapped to a color. Some connected components only contain one contact, whereas others several electrodes.*

6.1.2 Identification of Intersecting Electrodes

In the previous subsection, a mask of the intracranial electrodes, the EMSK, was extracted from the patient's CT. This subsection presents a step performed to identify close or intersecting electrodes in the mask, used to speed up computations and increase the accuracy of later stages of the pipeline.

The entry points and angles of insertion of the electrodes vary between all patients, to maximize the chances of successfully locating their EZ. Sometimes, particular regions of the brain are suspected, resulting in several electrodes implanted near these regions. But some other electrodes are sometimes implanted apart from the others.

In the EMSK, the content of a CC is undetermined; it can either contain contacts from the same electrodes, or contacts from different electrodes. This uncertainty is also valid for the relationship between two close CCs; they can contain contacts from the same or different—intersecting—electrodes, as illustrated in figure 6.4. But, around the intersection between electrodes, metal artifacts can provoke a bigger CC to appear, shaped like a ball and encompassing several contacts from different electrodes at once. Therefore, determining the nature of that relationship—same or different electrodes—can be hard. However, reversing the logic provides a more interesting observation; two distant electrodes in the initial CT scan always produce CCs in the EMSK

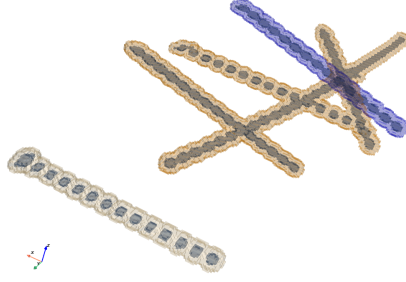


Figure 6.5: A 3D view of the transparent, colored post-dilation connected components (DCCs) from a subset of electrodes of Patient 1’s CT scan. Each color represents a DCC. The opaque, gray voxels are the original electrode mask. Dilation was performed with a structuring element of radius $R = 2$. In the 3D viewer, the blue and orange DCCs are visibly not intersecting.

that are also distant from each other. The following paragraphs detail how this observation is exploited to uncover relationships between the electrodes.

Firstly, the EMSK is dilated using a spherical structuring element of radius R . The impact of the choice of R is discussed below. The dilation has two effects:

- It erases the potential holes between the contacts of a given electrode, even if they were separated into different CCs in the EMSK. All the contacts of an electrode are then guaranteed to be joint.
- It joins close or intersecting electrodes.

After dilating the EMSK, the resulting connected components are identified and labeled using integers in $\{1, \dots, N_{cc}\}$, where N_{cc} is the number of post-dilation connected components (DCC). For clarity, the labels will be referred to as the post-dilation connected components identifiers (DCCid). Figure 6.5 illustrates the DCCs obtained by dilating a subset of CCs of Patient 1’s CT scan. As desired, voxels or CCs that come from the same electrode are guaranteed to belong to the same DCC (*i.e.* possess the same DCCid), while voxels or CCs with different DCCids are guaranteed to belong to different electrodes. However, some CCs belong to the same DCC, but different electrodes, as showed in the orange DCC in the figure.

In the dilation process, when choosing the hyperparameter R , the radius of the structuring element, it is crucial to use a value large enough to ensure that all contacts of a given electrode are joined. Otherwise, the disconnected sets of contacts will be treated as separate electrodes in the rest of the pipeline, and the algorithm may fail. This will be discussed in greater detail in section 7.4.2. Fortunately, in CT images with an in-slice resolution of 1 mm and electrodes with an inter-contact distance of less than 4 mm, the voxels are in practice almost never separated by more than 2 or 3 voxels. Hence, the choices $R = 2$ or $R = 3$ are sufficient. The latter was set as default, as it proved to be functional with the images in the study’s dataset. Higher values of R could also be considered, in order to guarantee the union of contacts. However, increasing R too much causes not-so-close electrodes to become joint. Hence, as R grows, the information gained from the centroids’ DCCids decreases. The theoretical value $R = \infty$ is equivalent to ignoring DCCids altogether and dealing with all electrodes as a single set. The algorithm would still be functional, but would benefit from less information about the electrodes.

6.1.3 Conversion from Voxel Mask to Set of Coordinates

In section 6.1.1, a mask of the electrode voxels was obtained in the form of a 3-dimensional image. The following objective is to convert the foreground of this mask, *i.e.* a set of voxels, into a set of meaningful points that hold information about the contacts, called the centroids.

To do that, the binary nature of the EMSK provides access to the mathematical morphology tools described in chapter 3, particularly ultimate erosion.

The figures 6.3 and 6.4 presented earlier show that the contacts appear as small balls in the EMSK. Even contacts linked within the same CC are distinguishable due to the inter-contact concavities. These particular shapes make ultimate erosion suitable for separating them. The only hyperparameter required by that morphological operator is the structuring element used for the successive erosions of the foreground. For 3D images, the most used elements are typically a cube or 3-dimensional cross, which have the effect of eroding the foreground one pixel at a time from all directions, respectively with and without accounting for diagonals.

Ultimate erosion applied to the EMSK—*i.e.* the CCs—returns a subset of voxels of that EMSK. Ideally, the subset would consist of multiple blocks of voxels, each representing one contact in the mask. Then, the center of mass of each block could be computed, outputting 3-dimensional coordinates: the centroids.

Figure 6.6 shows the blocks actually returned by ultimate erosion of Patient 1’s EMSK with a 3D cross structuring element. Some electrodes such as B and E produce several connected components, each represented by a point (*i.e.* centroid) in figure 6.7. However, most electrodes produce a significantly lower number of components than there are contacts, such as A, C and D. This unexpectedly low number of centroids can generate problems later in the pipeline, when trying to regress electrode models from these.

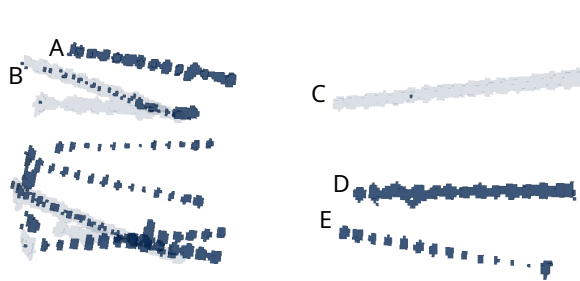


Figure 6.6: A top view of the result of ultimate erosion (opaque) on Patient 1’s electrode mask with a 3D cross structuring element, and the transparent input.

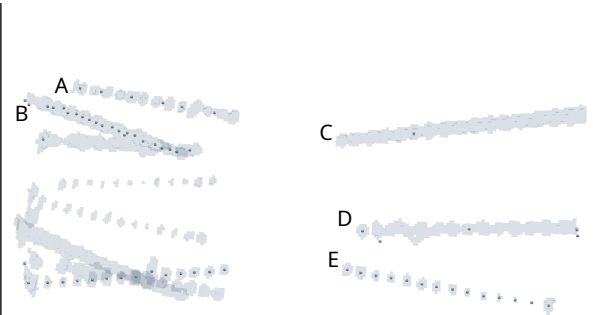


Figure 6.7: The centers of mass (points) of the connected components in figure 6.6, over the transparent initial electrode mask

To investigate the cause of this phenomenon, isolating one electrode and viewing it from the side can help. Figure 6.8 zooms on electrode C, and compares it to the structuring element used. Due to the high slice thickness (given in table 5.1), all regions in the electrode—except one—are two voxels high in the image. However, the 3D cross is three voxels high. Hence, the almost entirety of the electrode is erased during the erosion step of the first iteration of ultimate erosion. Since all the contacts are still joined when they are erased, the ultimate erosion algorithm considers them as a single connected component, and hence they collectively produce only one centroid. Since it is bigger than the 3D cross, the choice of the commonly-used 3D cube as the structuring element is also ruled out, because it would result in harsher erosion, and thus yield even less centroids.



Figure 6.8: A side view of Patient 1's mask of an electrode (blue), and a 3D cross for comparison (orange). The dot is at the position of the only voxel that survives the first erosion

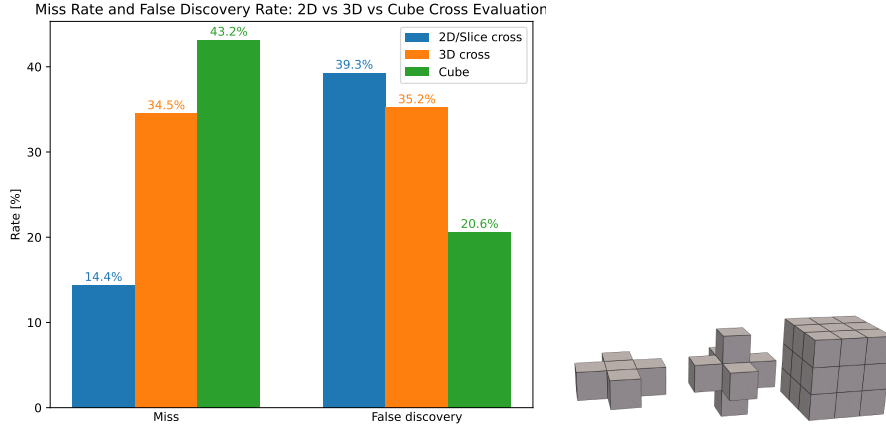


Figure 6.9: The proportions of false negatives and false positive rates in the centroids computed by ultimate erosion using several structuring elements. Some centroids that are too far away from a contact can be counted as both a contact missing where the centroid is supposed to be, and a false discovery where it actually is. The values were computed on all patients, with respect to their ground truths. The three structuring elements are displayed on the right (in-slice cross, 3D cross, cube).

To remedy this problem, a 2D cross that spans in the axial (*i.e.* XY) plane was used as the structuring element. The idea behind such a choice is to locate the contacts within the slices separately. This modification takes advantage of the resolution of the image, which is better within than across slices in most CT volumes, as it is the case here (see section 5.1). Hence, cavities between contacts are more distinguishable. Furthermore, beam-hardening (see section 1.4) tends to naturally separate the contacts in a slice into different CCs. Figure 6.9 plots the efficiency of the three structuring elements mentioned. As seen before, the 3D structuring elements tend to miss more centroids. On the other hand, the in-slice—*i.e.* 2D—cross solution seems to produce more centroids than expected. As a reminder, the objective of this centroid extraction module is to retrieve coordinates that bear information about where the contacts are located in the CT scans. In that sense, false positives can be filtered out later in the pipeline, but false negatives constitute permanent loss of information about the contacts, which will hinder the modeling of electrodes in the following steps of the global pipeline. Hence, it is more important to avoid false negatives than false positives.

Figure 6.10 displays the result of ultimate erosion with an appropriate in-slice cross used as structuring element. Almost all contacts are successfully assigned with a centroid. Excessive centroids can be seen at intersections, and in electrodes with a significant angle of insertion. The latter can be explained by the repeated detections of a single contact across several slices. Sections 6.2.1 and 6.2.3 will discuss the effect of the number of centroids on the rest of the algorithm.

To optimize the process described in this section, ultimate erosion is performed separately for all

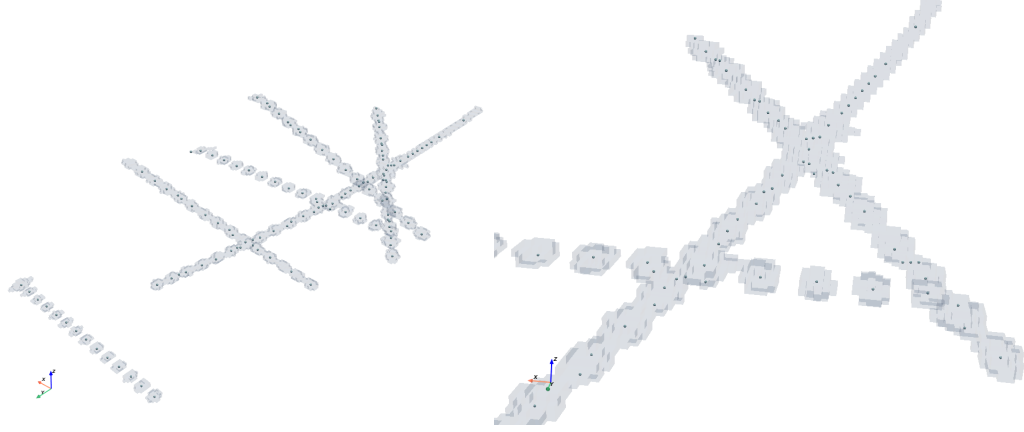


Figure 6.10: *The result of ultimate erosion on Patient 1’s electrode mask with the in-slice cross structuring element. The image on the right zooms on some intersections.*

DCCs. Before each application, the EMSK is cropped to the smallest window that fully contains the DCC considered. This small modification allows minimizing the size of the images handled by ultimate erosion, significantly speeding up the computations. Additionally, centroids obtained from the ultimate erosion of a DCC are labeled with the equivalent DCCid. The upcoming section 6.2.3 presents how centroid DCCids helps increase the accuracy and reduce the time complexity of the algorithm.

6.1.4 Summary of the Centroid Extraction Module

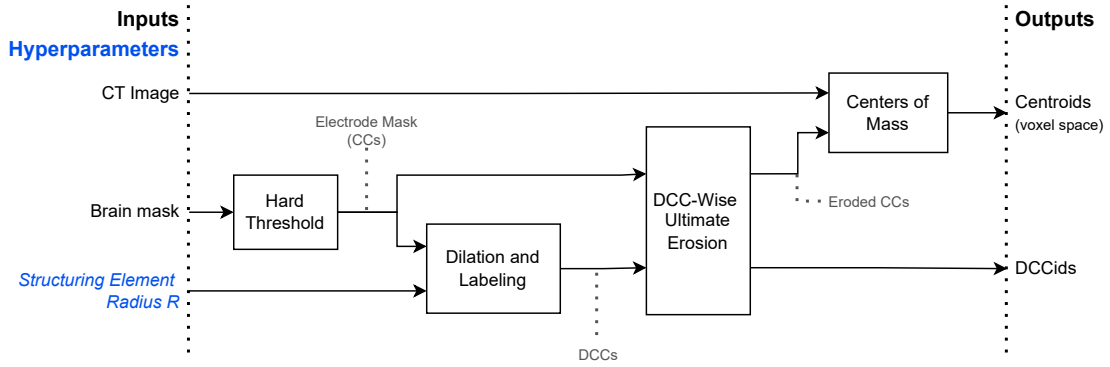


Figure 6.11: *The summary of the centroid extraction pipeline. It leverages the connected components (CCs) in the thresholded brain mask to compute centroid coordinates and identify their respective group of electrodes, expressed as a post-dilation connected component identifier (DCCid).*

The flow chart in figure 6.11 summarizes the processes presented in this section. First, an EMSK is computed from the CT scan and brain mask. It is then dilated to identify groups of intersecting or close electrodes, and label them with DCCids. Following that, ultimate erosion is separately applied to each DCC to compute a set of centroids and tag them with the equivalent DCCid. Finally, all centroids are reunited into a single set.

Table 6.1 details the inputs, outputs and hyperparameters of the centroid extraction module.

Table 6.1: *Summary of the centroid extraction module’s inputs, outputs, and hyperparameters.*

Type	Name	Description
Inputs	CT	The grayscale 3D volume to process
	Brain mask	A binary 3D mask of the brain in the CT
Outputs	Centroids	List of 3D points approximating the contacts
	DCCids	Identifier of the set of electrodes of each centroid
Hyperparameter	$R := 3$	Radius of the structuring element for computing DCCs

6.2 Linear Modeling of the Electrodes

In the previous section, a set of centroids was extracted from the CT volume, each associated with a 3-dimensional position in the voxel space, and a DCCid that reflects the group of electrodes it belongs to. This section presents a module that computes linear approximations of the electrodes using this data. It also requires as a hyperparameter the number of electrodes in the CT image.

This module first assumes that the electrodes in the patient’s head are either straight or curved with a reasonably low curvature (angle of less than 30° between the two ends). To begin with, all electrodes are approximated by straight lines. Curvatures are handled in the post-processing module described in section 6.3.

Before continuing, the centroids are first converted to physical space using the CT’s affine matrix, so that the distance between two centroids is isotropic with respect to the real world, as seen in section 1.3

6.2.1 Estimating the Inter-Contact Distance

To generate sequences of equidistant contacts later in the pipeline, the inter-contact distance must be known. To avoid requiring additional input from the user, it is estimated from the centroids computed in section 6.1.

First, the distance matrix $\mathbf{D} \in \mathbb{R}^{n \times n}$ of the n centroids is computed. Each element $\mathbf{D}_{ij} := \|\mathbf{c}_i - \mathbf{c}_j\|$ stores the euclidean distance between two centroids $\mathbf{c}_i$ and $\mathbf{c}_j$ with $i, j \in \{1, \dots, n\}$. The preceding conversion of all coordinates to physical space fortunately makes this distance isotropic with respect to the real world. The diagonal is then invalidated by forcing $\mathbf{D}_{ii} := \infty \forall i \in \{1, \dots, n\}$.

Then, for each centroid, the two shortest distances to other centroids are retrieved. Having invalidated the diagonal in the previous step prevents the centroids from picking themselves as one of their close neighbors. All the retrieved values are then concatenated into a single list.

The inter-contact distance is defined as the statistical mode of the distances between pairs of neighbors. To find it, the density function of the inter-neighbors distances is estimated using Kernel Density Estimation [59] with a Gaussian kernel of standard deviation 0.1 mm. This value was chosen as it provided a good trade-off between the smoothing of the spikes in the density function, and the preservation of the main maxima. Figure 6.12 illustrates the summary of the process for Patient 2. From the scattered neighbor distances stored in the list, Kernel Density Estimation is performed, and the distance at its global maximum is retained as the inter-contact distance. In the figure, the second highest peak is caused by excessive centroids, closer to their neighbors than actual contacts are. Hence, this subsection requires the hypothesis that the peak formed by noisy centroids is lower than that of actual neighbors, *i.e.* that the number of excessive centroids stays moderate. In the figure, it can be observed that this is

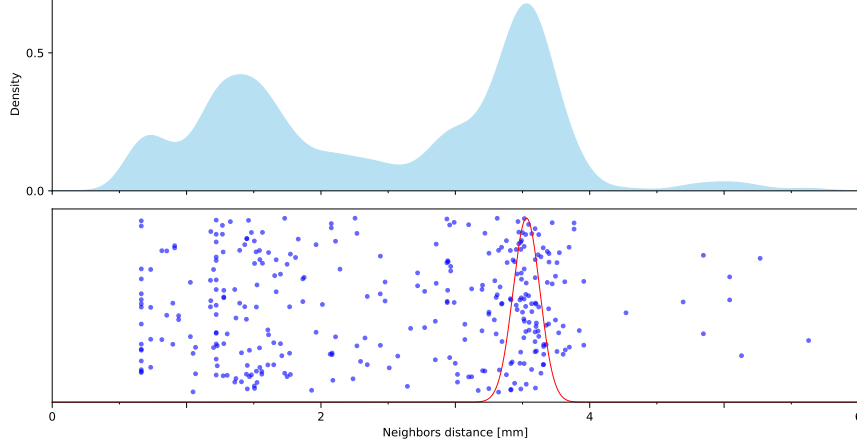


Figure 6.12: (Top) The result of the Kernel Density Estimation (KDE) of the inter-contact distance of Patient 2’s electrodes. (Bottom) A jitter scatter plot of the neighbor distances used to estimate the top plot. The red gaussian shows the moving kernel used by KDE, centered at the global maximum of the density function.

the case, despite the false positive rate of approximately 40% caused by the way centroids are extracted—as shown in figure 6.9 and explained in section 6.1.3—, this rate does not constitute an issue for the correctness of the estimated inter-contact distance.

6.2.2 Identifying Points of Interest among the Centroids

There are various ways for estimating linear electrode models from the cloud of centroids previously computed. For example, if each centroid’s EID were known, the parameters of the models could be computed through EID-wise linear regression of the centroids.

However, EIDs are unknown at this stage of the algorithm. To find them, the following approach is used:

1. Identify the two ends of the electrodes.
2. Approximate the axes of the electrodes by joining the ends of each with a line segment.
3. Classify the centroids to the closest segment to obtain the electrode labels.

Then, the linear models could be computed from the centroids and these labels. This subsection details how candidates for the tips are detected among the centroids, while the following subsection 6.2.3 explains how the actual tips are selected among these candidates to find optimal models.

For clarity, the centroids that constitute candidates for the tip of an electrode will be referred to as Points of Interest (POIs).

To extract POIs, the list of centroids is first re-expressed as a set of fully-connected graphs, each defined as:

- **Nodes:** all the centroids with a given DCCid;
- **Edge weights:** the real-world distance between two nodes/centroids.

Then, Kruskal’s algorithm (see algorithm 3) is applied onto each graph separately to extract its minimum spanning tree. By definition, all electrodes within a tree—*i.e.* within the same

DCC—are connected by at least one edge to one of the others. All trees are then merged into one sparse graph that contains all the centroids and one connected component per DCC. The reason for computing different graphs and trees is to ensure that electrodes in disconnected DCCs also stay disconnected in the final graph.

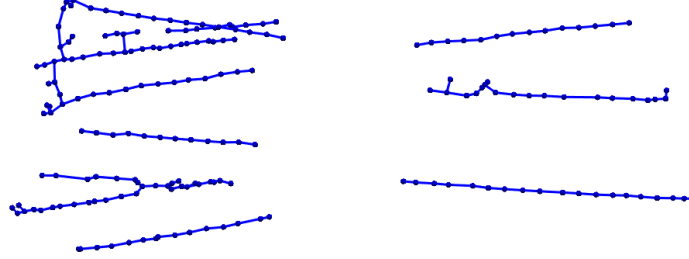


Figure 6.13: *The aggregation of the minimum spanning tree of each post-dilation connected component (DCC) using Kruskal’s algorithm for Patient 1. The nodes are the centroids. The electrodes linked in the graph belong to the same DCC, and electrodes in different DCCs stay disconnected in the graph.*

Figure 6.13 illustrates the graph obtained by applying the procedure described above on a subset of Patient 1’s DCCs. Some longer trails in the graph visibly represent the initial electrodes. The next objective is to identify centroids that are good candidates for being at the ends of such paths, *i.e.* identify the POIs. By looking at the trees on the right, choosing the leaves of the tree as the candidates for the electrode tips could seem like a reasonable choice. But the tree on the top left shows that the choice for good candidates is not as trivial; the tip of the most top left electrode is not a leaf. Therefore, some additional rules must be established to effectively retain the electrode tips as POIs.

The set of POIs is defined as the aggregation of three types of nodes:

- Leaves;
- Neighbors of 3-way intersections, *i.e.* neighbors of nodes of degree 3;
- Nodes of degree 2 whose edges form a sharp angle, defined as a change of direction greater than 45° between two successive edges.

The reason for not considering all centroids in the graph as POIs by default is discussed in the following subsection 6.2.3. Figure 6.14 illustrates some trees with highlighted POIs. Typically, trees that consist of a single electrode (the two trees at the top) contain 2–8 POIs, depending on how noisy some nodes are, whereas trees with intersecting electrodes or strong centroid noise can have up to a few dozens (the two trees at the bottom). Using these rules, it can be observed on the figure that all electrode tips were effectively identified as POIs. The value of 45° that defines sharp angles was chosen as it showed a satisfactory trade-off between precision, *i.e.* detecting actual electrode intersections, and recall, *i.e.* ignoring the angle formed by two actual contacts and a noisy centroid between them.

6.2.3 Computing an Optimal Set of Linear Models using Search Graphs

Justification and Definition of the Graph Search Approach

The previous subsection extracted a set of candidates for the tips of the electrodes: the POIs. To compute the model of a straight electrode, all that is needed is to select a pair of POIs and

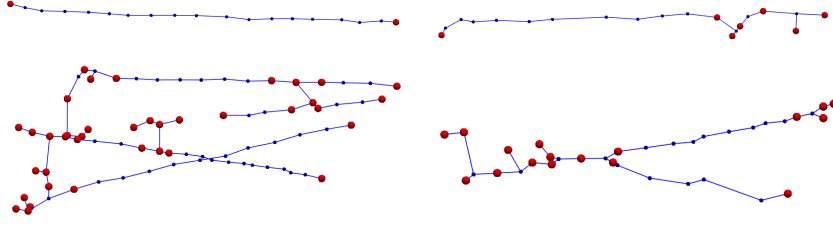


Figure 6.14: Four examples of trees and, in red, their Points of Interest (POIs). The two trees at the top each contain one electrode. The trees at the bottom contain respectively four and two electrodes. All electrode tips have successfully been identified as POIs. The number of POIs seems to grow with the number of electrodes.

join them by a line segment. Using the hyperparameter K , the total number of electrodes in the CT scan, the objective of this subsection is to find an optimal group of K electrode models, *i.e.* the group of electrodes that best fits the centroids. This problem is equivalent to finding an optimal combination of K pairs of POIs. Therefore, names such as " K -set of models" or "groups of K pairs of POIs" can be used interchangeably in this section. The line segment models used and some related concepts are defined later in this section.

To determine the K most optimal pairs of POIs, the most straightforward approach would be to compute all possible pairs of POIs, then all possible combinations of K pairs. From each combination, the associated group of K linear models would be computed, then evaluated by a function that scores how well they fit the data. The optimal group of models would then be the one that yields the highest score.

Though this method is intuitive and can easily be implemented, it possesses a major drawback. Using combinatorics, it is possible to compute the number of possible K -sets of models in a DCC with N POIs, which is equal to

$$\text{Nb groups in DCC} = \frac{N!}{(N - 2K)! \cdot 2^K \cdot K!}, \quad (6.1)$$

where $p!$ denotes the factorial of p . To give an idea of how impractical this number is, here are some examples. In a DCC with 4 electrodes and a reasonable number of 18 POIs, accounting for the intersections and some sharp angles, there would be 4,594,590 possible groups of models. Even with the generous assumption that 10,000 groups can be assembled, regressed, and scored every second, it would take 7 minutes 40 seconds to process them all. In the unlucky but possible case where 32 POIs are found within the DCC, it would take 30 hours to evaluate all 1,104,421,500 possible groups! Therefore, computing all possible combinations of pairs is not a reasonable solution for the problem at hand.

To address this issue, this paper proposes viewing the set of all configurations of groups as a graph, such that the BFS algorithm (defined in section 4.2) can be applied to retrieve the best one. The concepts used for the search were defined as follows:

Node: each node in the graph represents a group of k pairs of POIs ($k \leq K$). A given POI can be used by at most one pair. The nodes in the graph can be rearranged as layers, as shown in figure 6.15. A layer contains all the nodes with a certain group size k . The depth of a node is defined as k .

Children: the children of a given node P in layer k are all the nodes in layer $k + 1$ that can be obtained by adding one pair to the group of P . The new pair must not feature a POI already

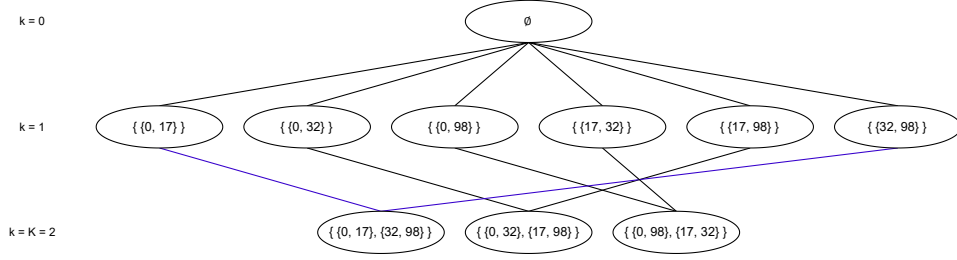


Figure 6.15: An example of search graph for a set of four points of interests (POIs) with centroid identifiers $\{0, 17, 32, 98\}$, and a limit of $K = 2$ electrodes. Each node represents a set of $k \in \{0, \dots, K\}$ pairs of POIs, which itself is equivalent to a group of k line segment models. The blue edges highlight the possibly multiple parents of a node.

used in P . Each node can have several parents, as shown in figure 6.15. For example, if A , B and C denote pairs of POIs, then node $\{A, B, C\}$ can be obtained through the parents $\{A, B\}$, $\{A, C\}$, and $\{B, C\}$. When computing the children of a node, only pairs of POIs that come from the same DCC must be considered. Hence, computing appropriate DCCs in the centroid extraction module (see section 6.1.2) can significantly reduce the number of pairs to consider, thereby reducing the number of children per node and, consequently, the overall size of the graph to search.

Target nodes: the nodes of interest for the pipeline are all the nodes with K models, *i.e.* nodes in layer K .

Evaluation function: the score of a node P represents how well the group of models of P fits the centroids. The evaluation function is detailed later in this section.

Figure 6.15 illustrates the full search graph of a scenario with a single DCC containing 4 POIs $\{0, 17, 32, 98\}$ and $K = 2$ electrodes. Each pair of POIs is represented as a 2-set of centroid identifiers. As stated above, nodes can have multiple parents, and nodes at depth K cannot have children. In the figure, it turns out that the bottom-layer node cannot have any children anyway, as there are no more unused POIs from which to assemble new pairs. But even if there were, no new children beyond K could be computed. The figure presents a small graph for clarity, but regular graphs obtained from 10–40 POIs typically possess layers with thousands or even tens of thousands of nodes.

Finding an Optimal Model in the Search Graph

With this search graph representation, the first approach presented above—which consisted in computing and evaluating all possible K -groups of models—is equivalent to iterating over all the nodes in the deepest layer of the graph. However, as seen previously, this layer grows factorially with the number of POIs.

This thesis proposes an approach that applies a modified version of the BFS algorithm (defined in section 4.2), such that only the best B children of each node are visited, where B is a hyperparameter called the branching factor. To implement that, during the expansion of a node, all of its children are computed, then scored using a metric defined later in this section. The children are then sorted by decreasing score, and the B best ones are returned, for the BFS algorithm to visit later. After visiting all nodes in layer $K - 1$, the best-scoring child in layer K is retrieved and returned as the most optimal group of K models.

Optimizations Introduced to the Search

To speed up the computations, three optimization enhancements were introduced to the BFS algorithm.

Firstly, since nodes can have multiple parents, one promising node is likely to be visited several times, from different paths of the graph. This results in duplicated paths in the graph, which significantly affect the algorithm’s speed. To avoid that, the list of all nodes to visit in a given layer was implemented as a set, such that all nodes can be visited at most once.

Secondly, during the expansion of a node, all of its children are evaluated. Again, since one node can possess several parents, that means that most children will be evaluated multiple times throughout the algorithm. To minimize the number of calculations performed, a cache system was implemented. When evaluating a child for the first time, the evaluation function—which requires several steps of computations—is called and returns the child’s score. That value is cached such that, the next time the child is evaluated from a different parent, the score is simply retrieved from the cache, which avoids requiring to perform the evaluation computations again.

Finally, in BFS, forcing decisions that are certain to be correct early in the search can significantly reduce the size of the graph searched and thereby increase efficiency without impacting the final result. To optimize the search, the algorithm was designed to prioritize the modeling of electrodes that are assured to be alone in their DCC, *i.e.* prioritize DCCs that contain exactly two or three centroids (two ends, one noisy at most). A DCC with four centroids or more centroids would be undetermined; it could contain a single electrode and noisy centroids, or the tips of several distinct electrodes. This optimization enhancement has the effect of significantly shrinking the size of the tree to search, as illustrated in figure 6.16.

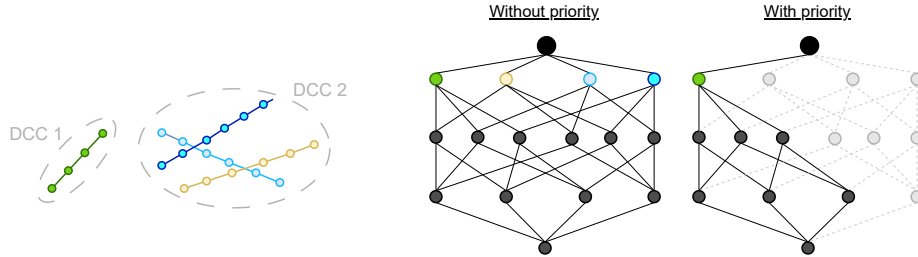


Figure 6.16: *On the left, a set of electrodes to model, lying in different DCCs. On the right, the search graphs used to model all four electrodes, with and without prioritizing the green one. For simplicity, each edge represents an additional correct electrode, not an arbitrary pair of centroids, as done in the pipeline. The actual benefits of applying priority are far greater in the latter case.*

Building and Evaluating the Models

The evaluation function used to compute the score of a node is defined with the following method. The k pairs of POIs in the node are used to compute time-parametrized line segments that join them. Hence, the line segment of the i -th model ($i \in \{1, \dots, k\}$) with assigned POIs $\mathbf{c}_{i,0}$ and $\mathbf{c}_{i,1}$ is defined as:

$$m_i := \{\mathbf{v}_i t + \mathbf{p}_i \mid t \in [t_a, t_b]\}, \quad \mathbf{p}_i := \mathbf{c}_{i,0}, \quad \mathbf{v}_i := \frac{\mathbf{c}_{i,1} - \mathbf{c}_{i,0}}{\|\mathbf{c}_{i,1} - \mathbf{c}_{i,0}\|}, \quad (6.2)$$

where $t_a = 0$ and $t_b = 1$ are the respective time parameters of $\mathbf{c}_{i,0}$ and $\mathbf{c}_{i,1}$ along the model’s curve. The directional vector $\mathbf{v}_i \in \mathbb{R}^3$ and anchor point $\mathbf{p}_i \in \mathbb{R}^3$ are the line parameters of the

segment, such that $\|\mathbf{v}_i\| = 1$. The symbol $\|\cdot\|$ denotes the standard euclidean norm. This definition is illustrated by figure 6.17.

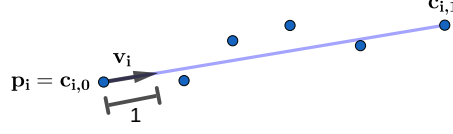


Figure 6.17: *Illustration of the definition of line segment models.*

Once all the models have been built, the following step is classifying all centroids according to their closest respective model. For that reason, the distance between all combinations of the k models and N centroids is computed and stored in a matrix $\mathbf{D} \in \mathbb{R}^{N \times k}$. The formula used for the distance between a centroid and a model depends on whether the projection of the former onto the latter lies within or outside the segment. An equivalent way of verifying this condition is computing the time parameter of the centroid projected along the curve, also called projection time, and comparing it to the segment's temporal range $[t_a, t_b]$. Figure 6.18 shows the way distances are computed between a segment and several points with different projection times. Through linear algebra, the projection time of any point $\mathbf{x} \in \mathbb{R}^3$ on the line of model m_i can be obtained as:

$$\text{proj_time}_{m_i}(\mathbf{c}_n) := \frac{(\mathbf{x} - \mathbf{p}_i) \cdot \mathbf{v}_i}{\|\mathbf{v}_i\|^2} \in \mathbb{R}, \quad (6.3)$$

which can be used to compute the distance (see figure 6.18) between any centroid $\mathbf{c}_n$ and model m_i as follows:

$$\mathbf{D}_{n,i} = \text{dist}(\mathbf{c}_n, m_i) := \begin{cases} \|\mathbf{c}_n - \mathbf{c}_{i,0}\| & \text{if } t < t_a \\ \frac{\|\mathbf{v}_i \times (\mathbf{p}_i - \mathbf{c}_n)\|}{\|\mathbf{v}_i\|^2} & \text{if } t_a \leq t < t_b \\ \|\mathbf{c}_n - \mathbf{c}_{i,1}\| & \text{if } t_b \leq t \end{cases} \quad (6.4)$$

Once $\mathbf{D}$ is filled, each centroid $\mathbf{c}_n$ is classified and labeled with the index of its closest model, called EID and defined as

$$l(n) := \arg \min_{i \in \{1, \dots, k\}} \mathbf{D}_{n,i} \quad \forall n \in \{1, \dots, N\}, \quad (6.5)$$

and the total score of the set of models $\{m_1, \dots, m_k\}$ in the evaluated node is computed as:

$$w_n := \exp\left(\frac{-1}{2} \cdot \frac{\text{dist}(\mathbf{c}_n, m_{l(n)})^2}{d^2}\right) \quad \forall n \in \{1, \dots, N\} \quad (6.6)$$

$$\text{score}(\{m_1, \dots, m_k\}) := \sum_{n=1}^N w_n, \quad (6.7)$$

where d is the inter-contact distance computed in section 6.2.1, and w_n is the contribution weight given to the centroid $\mathbf{c}_n$.

The scoring function (formula 6.7) counts the total number of centroids approached by at least one model of the group, where the contribution of each centroid to the count is weighted as a decreasing function of its distance to its closest model (formula 6.6). Figure 6.19 plots the

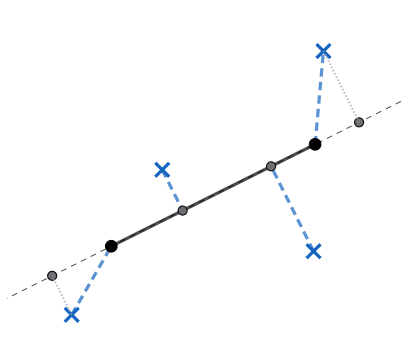


Figure 6.18: Distances (blue, dashed lines) between a line segment model (black) and several points (crosses), depending on their respective projection onto the segment's line (gray points).

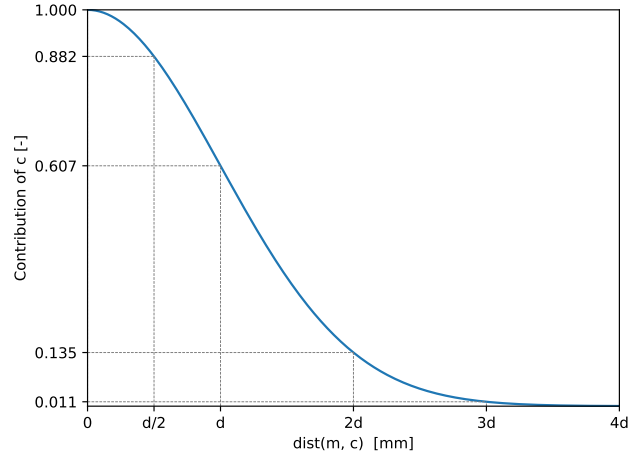


Figure 6.19: The weight of a centroid c as a function of its euclidean distance $\text{dist}(c, m)$ to a line segment model m . Parameter d is the average inter-contact distance.

evolution of a centroid's weight/contribution as a function of the distance to its closest model. Centroids that perfectly lie on their assigned model contribute to the score with a weight of 1, whereas centroids that are not sufficiently approached by any model bring a contribution close to 0. Furthermore, the contribution of a centroid to the score is only determined by its single closest model. This property discourages greedy behaviors such as fitting multiple models along a single electrode with many centroids, using multiple close POIs, to artificially increase the group's score.

For the graph search problem defined above, the effect of the evaluating function is to favor children whose added model fits the highest number of new centroids. Since centroids are located along the electrodes in the CT image, the best scoring nodes in a layer are those that successfully model long electrodes. And the best node in layer K is the one that successfully models all electrodes, *i.e.* fits all centroids.

Building the Final Line Segment Models

After the graph search algorithm ends, the best node found is returned. The line segments finally returned by this module are obtained the following way:

1. Build the line segment models from the node's pairs of POIs using formula 6.2.
2. Classify the centroids based on their closest respective model using formula 6.5. The set of centroids that are assigned to a given model are called its inliers.
3. Recompute all models m_i ($i \in \{1, \dots, K\}$) from their inliers:
 - (a) Perform linear regression on the inliers, and update the model's parameters $\mathbf{p}_i$ and $\mathbf{v}_i$ by those returned by the regression.
 - (b) Compute the projection time of all inliers onto the model using formula 6.3. Update the model's parameters t_a and t_b with the inliers' lowest and highest projection times respectively.

After repeating these three steps for all pairs in the optimal node, the module outputs the K updated segment models and the list of all centroids' EIDs, the identifiers of the model they were respectively assigned to.

6.2.4 Summary of the Linear Modeling Module

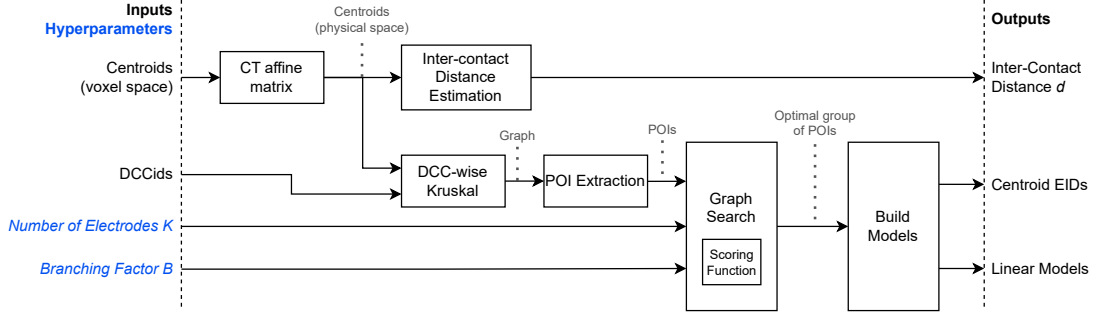


Figure 6.20: *Summary of the linear modeling module. The module accepts centroids and their post-dilation connected components identifiers (DCCids). It selects points of interest (POIs) among these centroids, then select the pairs of centroids that best represents the ends of the electrodes. It returns the estimated inter-contact distance, the linear models, and the centroids' electrode identifiers (EIDs).*

Figure 6.20 displays a detailed summary of the linear modeling module's pipeline. First, the centroids computed in section 6.1 are converted to physical space using the CT image's affine matrix to ensure isotropic distances. Then, their positions are used to estimate the inter-contact distance using kernel density estimation. In parallel, Kruskal's algorithm is used to compute centroid trees inside the DCCs, which are then used to identify POIs that are potential end points of the electrodes. Graph search is then performed to retrieve and build the group of segments built from POIs that best fit the centroids. Finally, these models are used to classify the centroids. Table 6.2 summarizes the inputs, outputs, and hyperparameters involved.

Table 6.2: *Summary of the linear modeling module's inputs, outputs, and hyperparameters.*

Type	Name	Description
Inputs	CT Affine	The conversion matrix from voxel to real-world space
	Centroids (vox.)	The positions of the centroids in voxel space
	DCCids	The DCC labels associated to the centroids
Outputs	d	The inter-contact distance in physical space
	Centroids (phys.)	The positions of the centroids in physical space
	Linear Models	A list of line segments that best fit the centroids
	EIDs	The result of classification of the centroids to the models
Hyperparameters	K	Number of models to generate
	B	Branching factor of the search algorithm

6.3 Post-Processing the Models

In the modeling module detailed in the previous section, a set of centroids contained in the electrodes was used to derive the optimal set of segments to model these electrodes, and a list

of EIDs to tie the centroids to their closest model. However, at this stage of the pipeline, two problems are still unresolved:

- Some electrodes can become slightly bent during or after their insertion into the patient’s brain. Line segments are not suited for modeling curved electrodes.
- The centroids constitute useful information about the overall position of the contacts within the image, but not their precise localization along the electrodes. This is partly due to a high false positive rate, as highlighted by figure 6.9.

The goal of the post-processing module presented in this section is to handle these points. It relies on two main steps:

1. Replace the linear models with curved ones.
2. Replace the approximate centroids by actually equidistant contacts along the curved electrodes.

This section delves into the details of the implementation of each post-processing step applied, to output a fitting set of contacts detected within the CT scan.

6.3.1 Curved Electrode Models

The first post-processing step is the replacement of the straight segment models obtained from the preceding module by curved ones. The underlying challenge is that electrodes can deform in various ways in the patient’s brain. Hence, being able to perfectly fit them all would require a large number of available curve models. But in practice, even the most bent electrodes keep a reasonable curvature and a stretched shape. In this module, curved electrodes are approximated by 3-dimensional parabolas.

In 3-dimensional geometry, a parabola is a particular type of time-parametrized curve that lies on a specific plane of $\mathbb{R}^3$, and resembles a 2D parabola within that plane. An example of a parabola is given in figure 6.21.

Appendix B delves into the details of a parabola’s mathematical definition. The takeaway from that appendix is that all parabolas can be uniquely identified by directional parameters, which themselves can be expressed in a compact way using a matrix $\mathbf{A} \in \mathbb{R}^{3 \times 3}$. Along time-parametrized curves, such as parabolas, points are associated to time values, also called time parameters.

Given a parabolic curve p represented by the matrix $\mathbf{A} \in \mathbb{R}^{3 \times 3}$, the coordinates of the point that lies along that curve at time $t \in \mathbb{R}$ are computed using the direct formula:

$$\mathbf{x}_p(t) = \mathbf{A} \begin{bmatrix} t^2 \\ t \\ 1 \end{bmatrix}. \quad (6.8)$$

Because electrodes do not have infinite lengths, the parabolic models used to represent them are restricted to a limited arc bounded by two time values t_a and t_b , such that $t \in [t_a, t_b]$. Figure 6.21 illustrates a time-bounded parametric curve.

Two additional functions are also declared here for later use. Their exact definitions are detailed in appendix B. These are:

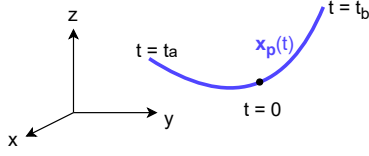


Figure 6.21: A simple example of time-parametrized parabolic curve defined in the range $t \in [t_a, t_b]$.

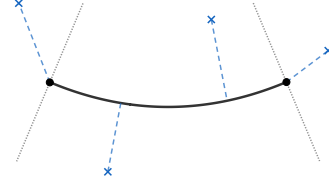


Figure 6.22: Distances (dashed lines) between a parabola arc and several points (crosses), depending on their respective projection time.

- $\text{proj_time}_p(\mathbf{x})$: returns the time of the projection of any point $\mathbf{x} \in \mathbb{R}^3$ onto the parabola p . This value can be outside the range $[t_a, t_b]$.
- $\text{dist}(\mathbf{x}; p)$: returns the distance between any point $\mathbf{x}$ and the parabola p , represented by matrix $\mathbf{A}$, as defined in formula 6.9. This function accounts for whether that point lies "outside" the parabola arc or not, as illustrated in figure 6.22. For points with a projection time contained in $[t_a, t_b]$, *i.e.* points between the gray lines in the figure, the distances to the parabola are returned. Otherwise, the distances to the closest end is returned.

$$t := \text{proj_time}_p(\mathbf{x})$$

$$\text{dist}(\mathbf{x}, p) := \begin{cases} \|\mathbf{x} - \mathbf{A}t_a\| & \text{if } t < t_a \\ \|\mathbf{x} - \mathbf{A}t\| & \text{if } t_a \leq t < t_b \\ \|\mathbf{x} - \mathbf{A}t_b\| & \text{if } t_b \leq t \end{cases} \quad (6.9)$$

From the received set of centroids, the modeling module returned an optimal set of linear models and the equivalent EIDs attributed to the centroids. To convert straight models to curved ones, the respective inlier centroids of each model are used to regress a parabola using the least squares method [60]. The contributions of the centroids to the regression are weighted as the function of the distance to their respective linear model, as defined previously. This weight function is the same as defined in formula 6.6.

The details of the regression are computation-heavy, and expounded in appendix B. The result of this section is a set of parabolic models, each matching the EID of the linear model it was derived from. The EIDs of the centroids remain unchanged through the process. The process is depicted for one model in figure 6.23.

It is worth noting that perfectly straight electrodes are not affected by this change to parabolic models. In such a case, the least squares method will naturally cause the resulting parabola to become flat, resulting in straight-looking curves.

6.3.2 Generating Equidistant Contacts

The second post-processing step to tackle is the replacement of the imprecise centroids by equidistant contacts along the curved electrodes modeled in the preceding section 6.3.1. However, as discussed earlier in the section 6.1, the number of centroids is not representative of the total number of contacts implanted along the electrodes; it tends to over-estimate and produce

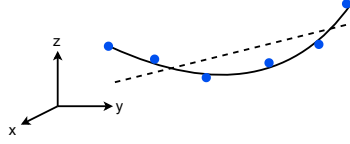


Figure 6.23: Example of replacement of a line segment model (dashed) by a parabolic one (solid) that better fits a given a set of inlier centroids (blue points).

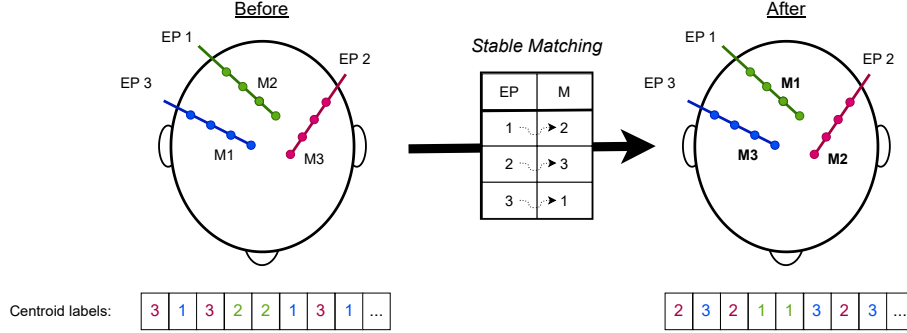


Figure 6.24: Illustration of the stable matching process. The identifiers of the models (M) are permuted to match those of the entry points (EP). The centroid labels ($EIDs$) are updated accordingly.

numerous false positives. Therefore, to generate the appropriate amount of contacts along each electrode, that amount must be known or found. For the early stage of development of the algorithm, requiring the exact number of contacts along each electrode was deemed sufficient, although it requires more information about the electrodes to provide to the algorithm. Therefore, the post-processing module presented in this thesis requires a list of the electrodes' entry points coordinates, and their respective number of contacts.

Matching the Models to their Number of Contacts

Until now, all models were considered as an unordered set; although they all had an identifier, used to label the centroids, their order did not matter. However, as the exact number of contacts along each electrode comes in the form of an ordered list, it is necessary to match the models to their respective electrodes. In other words, the identifiers of the models must be permuted and aligned with those in the list.

Table 6.3: An example of the list required to obtain the number of contacts to generate along each model. The gray text constitutes information inherent to the list, and does not have to be provided by the user.

<i>Electrode ID</i>	# Contacts	Entry coordinates (x, y, z)
1	16	(245, 97, 154)
2	9	(119, 195, 87)
...	...	...

To help the program with this task, additional information about the electrodes is given: the coordinates of their respective entry points. Table 6.3 summarizes the content of the parameters used by the post-processing module. The list is provided to the pipeline as a CSV file. The

coordinates are expressed in voxel space, then converted to real-world space, where the centroids are, using the CT image's affine matrix. The problem of matching the models to their respective electrode is illustrated by figure 6.24. It is performed in two steps:

1. computing the distances between all models and entry points;
2. determining the bijective re-labeling of the models based on these distances.

For the first step, a process similar to the one described by formula 6.4 was used. The distances between the electrode entry points $\mathbf{x}_1, \dots, \mathbf{x}_K$ and the parabolic models $p_1, \dots, p_K$ are stored in a matrix $\mathbf{D} \in \mathbb{R}^{K \times K}$, where

$$\mathbf{D}_{i,k} := \text{dist}(\mathbf{x}_i, p_k). \quad (6.10)$$

Once the distance matrix $\mathbf{D}$ is fully computed, the second step to perform is applying the Stable Matching Problem [61] (also called Stable Marriage Problem) to $\mathbf{D}$ to obtain a stable, bijective matching of close entry points and models. As a result, the EIDs of the models—and centroid labels—are permuted to match those of the electrodes.

After aligning the identifiers of the models and electrodes, the number of contacts to generate along each model is retrievable from the user-provided list.

Generating the Contacts

In this section, the imprecise centroids are replaced by points that accurately represent the contacts in the image.

For clarity, this section focuses on generating contacts along a single parabolic model p . The process described here must be repeated for all models. Section 6.2.1 estimated the inter-contact distance d , and section 6.3.2 retrieved the number of contacts n to generate along the model. Since p is time-parametrized, computing a sequence of contacts $[\mathbf{k}_0, \dots, \mathbf{k}_{n-1}]$ along p is equivalent to computing the sequence of time parameters $[t_0, \dots, t_{n-1}]$ of these respective contacts. Because the electrodes are assumed to have reasonable curvature—less than 30° between the tips—, equidistant contacts can be assumed to be equally distant in time, *i.e.* $t_{i+1} = t_i + \Delta t$, where $\Delta t \in \mathbb{R}$ is the inter-contact time. Hence, the objective becomes retrieving the sequence $[t_0, t_0 + \Delta t, \dots, t_0 + (n-2)\Delta t, t_0 + (n-1)\Delta t]$ that best fits the contacts in the CT image, where $t_0 \in \mathbb{R}$ is the starting time of the sequence. Retrieving the optimal sequence of points requires retrieving the optimal values for t_0 and Δt . The following paragraphs detail the steps performed to achieve this.

The length of a time-parametrized curve $\mathbf{x} : \mathbb{R} \rightarrow \mathbb{R}^3$ between two times t_a and t_b , also called arclength, is computed as follows [62]:

$$\text{ArcLength}_{\mathbf{x}}(t_a, t_b) := \int_{t_a}^{t_b} \|\mathbf{x}'(t)\| dt \quad (6.11)$$

As a reminder, the time parametrization $\mathbf{x}_p(t)$ is defined in formula 6.8. The inter-contact time Δt is obtainable by minimizing the absolute difference between the inter-contact distance and the parabola's arclength between two contacts at times 0 and Δt respectively, as follows:

$$\Delta t := \arg \min_{\Delta t' \in \mathbb{R}} \|d - \text{ArcLength}_{\mathbf{x}_p}(0, \Delta t')\|. \quad (6.12)$$

Once the optimal Δt has been determined, the next objective is to find the optimal starting time t_0 for the sequence. The proposed solution is to use an optimizer to select t_0 so as to

minimize the total sum of the squared distances between each generated contact and its closest centroid. That total sum acts as a loss function for the optimizer. The idea behind this solution is that, in most contacts of the CT image, a centroid will tend to be detected close to the center. Hence, these will be the closest centroids to the generated contacts, and the optimizer will try to position the sequence as to align the latter to the former. This strategy can be re-expressed mathematically as the following double optimization problem:

$$\begin{cases} \mathbf{c}_i := \arg \min_{\mathbf{c} \in C} \|\mathbf{x}_p(t_0 + i\Delta t) - \mathbf{c}\| & \forall i \in \{0, \dots, n-1\} \\ t_0 := \arg \min_{t \in \mathbb{R}} \sum_{i=0}^{n-1} \|\mathbf{x}_p(t_0 + i\Delta t) - \mathbf{c}_i\|^2 \end{cases}, \quad (6.13)$$

where C is the set of all centroids, $\mathbf{c}_i \in \mathbb{R}^3$ is the closest centroid to the i -th contact in the sequence, and $\mathbf{x}_p(t_0 + i\Delta t) \in \mathbb{R}^3$ is the coordinates of the i -th contact in the sequence.

Once both Δt and t_0 have been computed, the optimal sequence of n contacts can be generated along the parabolic model p as follows:

$$\begin{aligned} \text{Sequence}_p(t_0, \Delta t) &= [\mathbf{k}_0, \mathbf{k}_1, \dots, \mathbf{k}_{n-1}] \\ &:= [\mathbf{x}_p(t_0), \mathbf{x}_p(t_0 + \Delta t), \dots, \mathbf{x}_p(t_0 + (n-1)\Delta t)] \end{aligned} \quad (6.14)$$

6.3.3 Summary of the Post-Processing Module

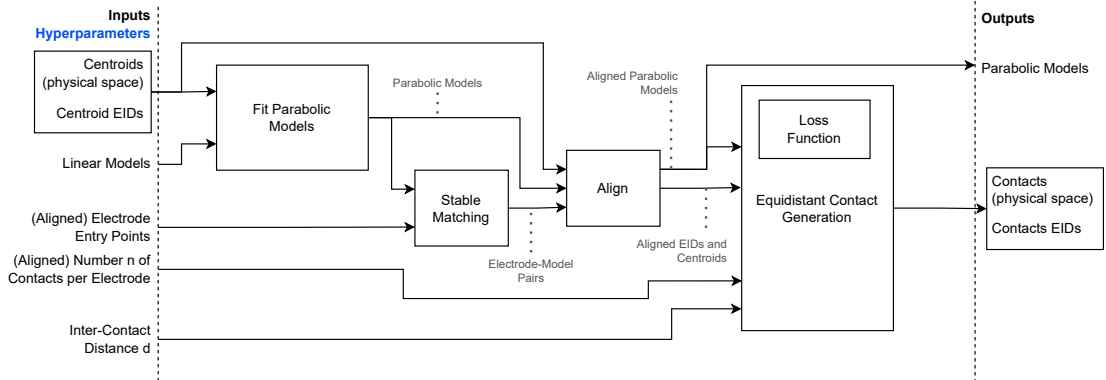


Figure 6.25: *Summary of the post-processing module. The module starts by replacing linear models by parabolic ones. Then, the models’ identifiers (EIDs) are permuted to match those in the hyperparameter electrode data. Finally, equidistant contacts are generated along each model.*

Figure 6.25 summarizes the pipeline followed by the post-processing module. First, the linear models computed in section 6.2 are refined into parabolic ones. Then, their EIDs are permuted to match those in a CSV file that specifies the entry points and number of contacts of all electrodes. Finally, the latter is used to generate equidistant contacts along the curved models. Table 6.4 specifies the module’s inputs and outputs.

6.4 Algorithm Summary

Figure 6.26 summarizes the flow of the entire algorithm. The CT image and brain mask are introduced to the centroid extraction module, which produces a set of meaningful points about the

Table 6.4: *Summary of the post-processing module’s inputs and outputs.*

Type	Name	Description
Inputs	Centroids (phys.)	The positions of the centroids in voxel space
	EIDs	The result of classification of the centroids to the models
	Linear models	A list of line segments that best fit the centroids
	Electrode Info	A CSV file that contains all electrodes’ entry points and numbers of contacts
	d	The estimated inter-contact distance
Outputs	Parabolic models	A list of parabolas that best fit the electrodes
	Contacts (phys.)	The positions of the predicted contacts in physical space
	Contacts EIDs	The electrode labels of the contacts

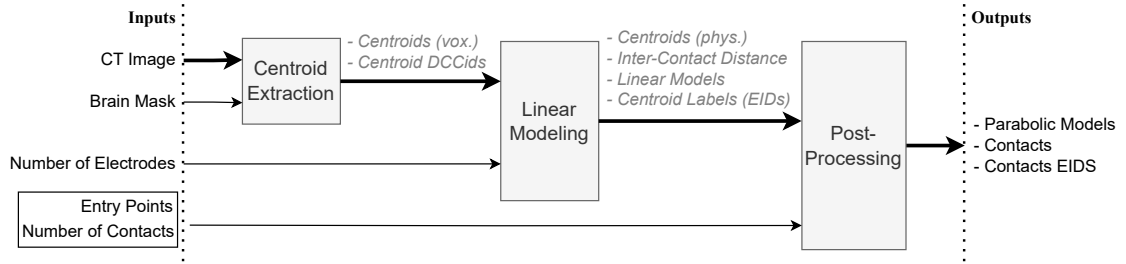


Figure 6.26: *Summary of the overall algorithm.*

electrodes (centroids) and identifies groups of intersecting electrodes (DCCs). This information is forwarded to the modeling module, which computes optimal line segments to model the electrodes and classify the centroids (EIDs). Additionally, it estimates the inter-contact distance. Finally, the post-processing module replaces the line segments by parabolas, to account for electrode bending, and generates optimal sequences of points along the parabolic models. These labeled points constitute the output of the algorithm.

The results are stored in a CSV file that contains, for all predicted contacts:

- its voxels and real-world coordinates;
- the label of its electrode, the EID;
- its position on the electrode, the PID.

6.5 Performance Assessment Metrics

To evaluate the performance of the algorithm, the ground truths presented in section 5.1 were used.

Centroid Extraction Module

No correctness metric was applied to the centroid extraction module. Its primary objective is to produce a set of meaningful centroids and their DCCids, for later stages of the algorithm to use. Since this objective is ill-defined, defining a metric for it is an arbitrary process, skipped here.

However, the impact of the dilation radius R —used to identify DCCs—on the execution time of the algorithm was studied by running the entire pipeline on Patient 1’s CT image with increasing

values of R . For each value, three runs were performed, to retrieve their respective means and standard deviations.

Linear Modeling Module

The coordinates of the ground truth contacts (GTC) were given as the set of input centroids of the module. Their respective DCCids were determined from a run of the centroid extraction module, using $R = 3$ and the EMSK obtained from the appropriate CT image. The modeling module was then run with $B = 2$ to model and classify the GTCs. The accuracy of the models was then computed as the percentage of GTCs that were classified into the correct electrode.

Additionally, the impact of the branching factor B was studied by running the isolated module on Patient 1's CT image with increasing B . For each value, five runs were performed. For each, the execution time of the module and the best score found—according to the metric in formula 6.7—were recorded.

Overall Algorithm

The performance of the algorithm was assessed by running all modules successively on all CT images, with the default hyperparameters $R = 3$ and $B = 2$. Each run returned a set of predicted contacts (PC). The sets of PCs and GTCs of each image were then compared; pairs of contacts were matched across the sets, if they were sufficiently close to each other and both unmatched. Figure 6.27 illustrates this process.

A true positive (TP) was recorded when a free GTC and a free PC were sufficiently close to each other. A false positive (FP) was recorded when a free PC was unmatched. A false negative (FN) was recorded when a free GTC was unmatched. The results were stored in table 6.5. For example, in the figure, there are two matched pairs ($TP = 2$), three free predicted contacts ($FP = 3$) and two free expected contacts ($FN = 2$).

Table 6.5: *Table of the values used to assess the quality of the contacts predicted.*

Number of cases	Contact predicted	No contact predicted
Contact expected	True Positive (TP)	False Negative (FN)
No contact expected	False Positive (FP)	—

The cell at the intersection of the row "no contact expected" and column "no contact predicted" is left empty, as analyzing the positions where no contact was predicted or expected does not yield relevant information about the result of the algorithm. To assess the quality of the predicted contacts, three metrics are used:

- Positive Predictive Value (PPV) $:= \frac{TP}{TP+FP}$. This metric gives the proportion of relevant contacts predicted. It is also called precision.
- Sensitivity $:= \frac{TP}{TP+FN}$. This metric gives the proportion of expected contacts that were successfully predicted by the algorithm. It is also called the recall.
- Distances between the contacts in the matched pairs.

The exact procedure followed for counting the number of TPs, FPs, and FNs is detailed here:

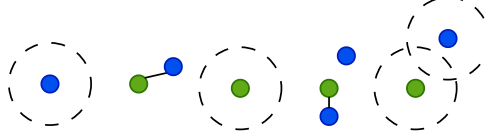


Figure 6.27: *Examples of matches (solid lines) between predicted (blue) and expected (green) contacts. Contacts can only be matched to other contacts in range if they are not matched yet. The dashed circle represents the maximal match range.*

1. The distance matrix between all pairs of GTCs and PCs was computed.
2. All values higher than 75 % of the inter-contact distance estimated from the GTCs were invalidated.
3. Stable Matching [61] was applied to the distance matrix, accounting for invalidated items (*i.e.* unmatchable pairs), to retrieve the list of GTCs and PCs matched. These were defined as the TPs.
4. All non-matched GTCs were defined as the FNs, *i.e.* missing contacts.
5. All non-matched PCs were defined as the FPs, *i.e.* excess contacts.

Chapter 7

Results and Discussions

This chapter elaborates on the results of the experiments conducted to assess the performances of the different stages of the algorithm presented in chapter 6. First, the performance of the algorithm and the impact of its hyperparameters are studied. The algorithm is then critically compared to other existing methods, and some of its limitations are explored. The chapter ends by proposing enhancements that could be provided to the algorithm in future works.

7.1 Performances of the Algorithm

7.1.1 Performances of Centroid Extraction

The process used to evaluate the impact of the dilation radius R on the algorithm is detailed in section 6.5. All tests presented in this section were performed on an Intel Core i7-10700×16 processor, on a single thread. The results, displayed in figure 7.1, show a strong correlation between R , the execution time of the algorithm, and the number of DCCs produced. This result seems reasonable. Significant spikes in the execution time occur when the number of DCCs drops, due to the increased number of pairs of POIs to consider in the search graph. The execution time also slightly increases with R when the number of DCCs is stagnant. This is most likely due to an unchanging execution time of the modeling module, and the increasing time required to dilate the EMSK in the centroid extraction module.

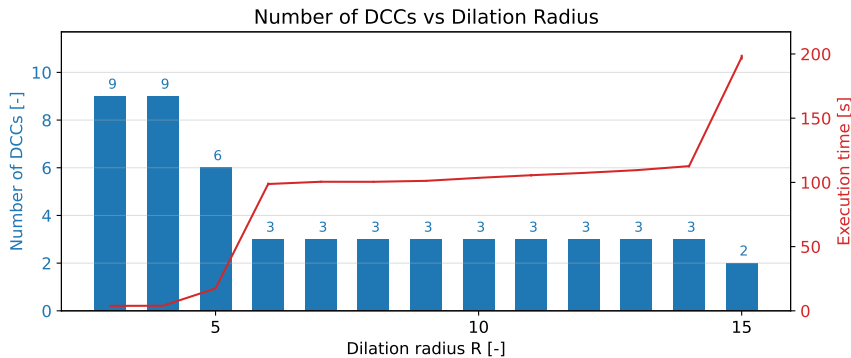


Figure 7.1: *Effect of the dilation radius R on the performance of the entire algorithm on Patient 1. Three runs were performed for each value R , and the standard deviations are indicated as vertical lines.*

7.1.2 Performances of Linear Modeling

Figure 7.2 illustrates the output models of the module executed with the inputs detailed in section 6.5. The module obtained an accuracy of 100% on both images, managing to perfectly classify all contacts from the generated models.

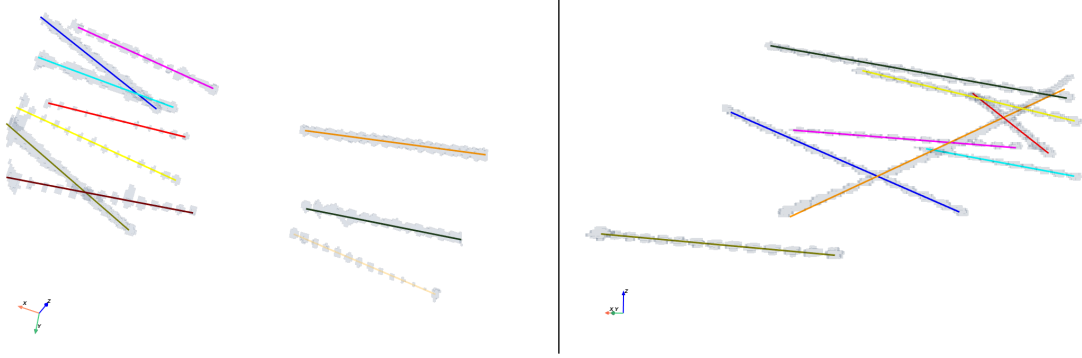


Figure 7.2: A 3D view of the contact-less linear models produced for patients 1 (left) and 2 (right). Some electrodes were removed from patient 1's image for clarity, but they showed similar results. The transparent electrode masks are shown for comparison.

Figure 7.3 summarizes the effect of the branching factor B on the module. It can be observed that a branching factor $B = 1$ is sufficient to find a high-scoring node in less than a second. Increasing the branching factor has the effect of exploring more nodes in the graph, resulting in slightly better scores returned, but significantly higher execution times. Therefore, values of 1 or 2 for the branching factor B prove to be sufficient for the the linear modeling module, to find high-scoring groups of models with low computational times.

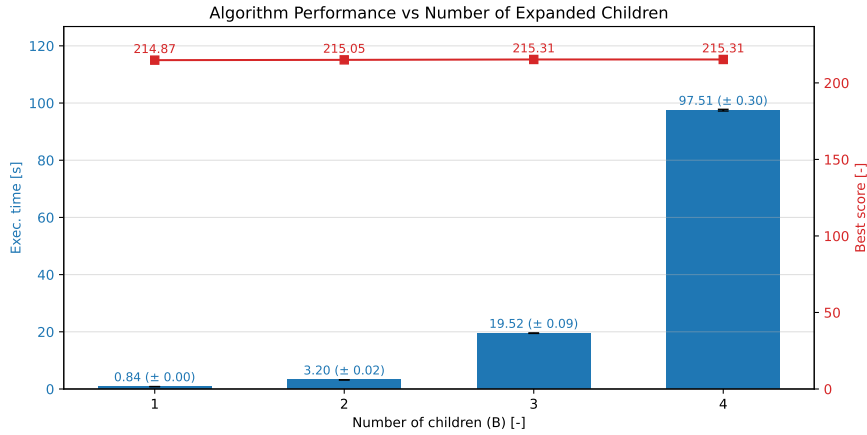


Figure 7.3: The execution time and best score found by the linear modeling module for several branching factors. The data was recorded on Patient 1's set of centroids. The execution times display both the mean and standard deviation measured over 5 runs.

7.1.3 Performances of the Algorithm

The process detailed to assess the performances of the algorithm are detailed in section 6.5. After processing the images of all patients and comparing the results to the ground truths,

the algorithm obtained the scores detailed in table 7.1. For each image, the distances between the predicted and expected contacts were, on average, smaller than the image’s resolution (see section 5.1). In other words, the average error is smaller than the size of a slice pixel, which constitutes satisfactory results. Figure 7.4 gives more detail about the distances between the matched pairs of PCs and GTs.

Table 7.1: *Summary of the results of the algorithm for both patients. The indicated metrics are respectively: the number of ground truth contacts (GTC), the Positive Predictive Value (PPV), the sensitivity, the mean error and standard deviation in millimeters, the 25% and 75% quartiles.*

Patient	# GTCs	PPV	Sensitivity	Mean error [mm]	[25%; 75%] [mm]
1	162	96.914	96.914	0.596±0.327	[0.379; 0.735]
2	116	98.276	98.276	0.410±0.221	[0.243; 0.531]

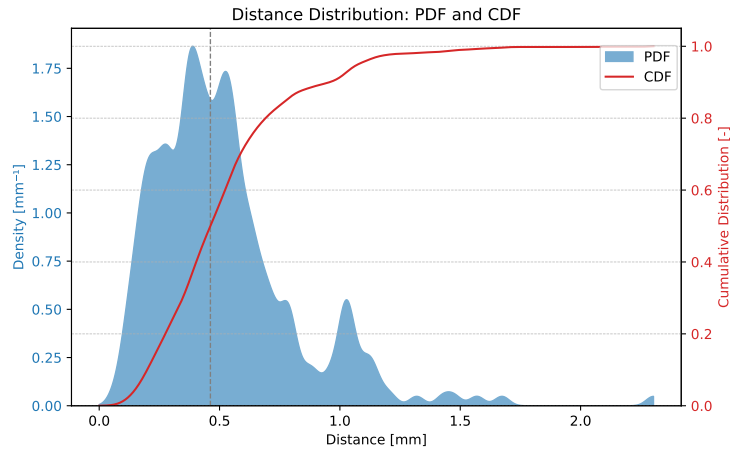


Figure 7.4: *The density (PDF, blue) and cumulated density (CDF, red) of the distance between the matched predicted and expected contacts in both images. The gray, vertical line is the median.*

The cause of the loss of PPV and sensitivity is illustrated in figure 7.5. Here, the parabolic models can be seen fitting correctly the electrodes. However, some electrodes present an unexpected shift of the generated contacts (section 6.3.2) by a one-contact or two-contact distance along their models. Each shift causes simultaneously a false positive and a false negative on the ends of the affected electrode.

Despite the shifts, the figure shows that the algorithm has successfully predicted contacts around the centers of the electrode mask contacts. Furthermore, the inter-contact distance estimated in section 6.2.1 and used to generate equidistant contacts shows satisfactory results, when compared to the electrode mask.

7.2 Comparison to Existing Methods

This algorithm constitutes a novel approach for locating sEEG contacts in CT scans. Out of all the existing methods, it is the only one that, to the author’s knowledge, makes use of graph theory and graph search to model the electrodes and find optimal groups of electrodes. Furthermore, though some existing methods already used mathematical morphology to process the thresholded CT image, this algorithm is the first one to use morphological operations to derive information about the positions of the contacts, through ultimate erosion.

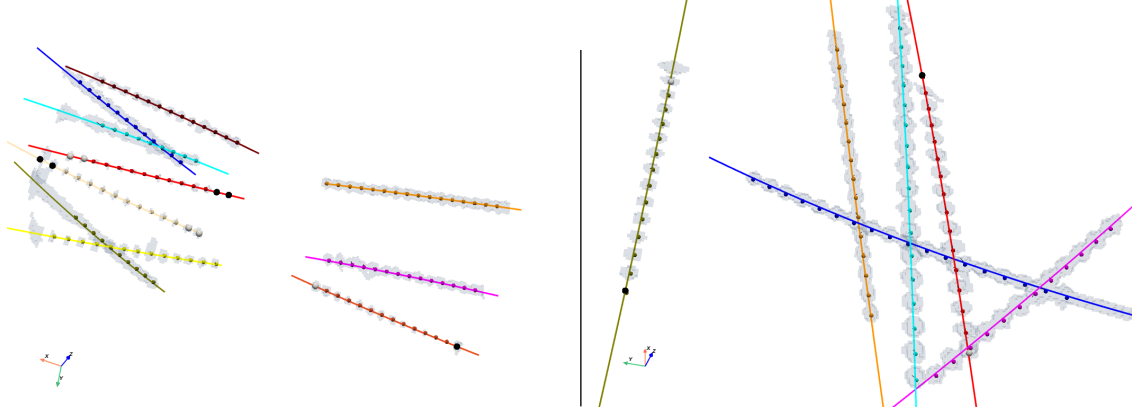


Figure 7.5: A 3D view of the contacts and models produced by the overall algorithm for patients 1 (left) and 2 (right). Black points represent false positives, and white points false negatives. Some electrodes were removed for clarity, but they didn't possess any misplaced contact. The transparent electrode masks are shown for comparison.

Table 7.2 compares the inputs and features of the method proposed in this thesis and those of the existing tools. Compared to the other methods studied [13]–[21], the algorithm proposed requires more prior information about the electrodes and CT image. On the other hand, it is the only one that considers and shows robustness to electrode bending, crossing electrodes, and strong CT artifacts simultaneously.

Furthermore, the program that runs the algorithm can be deployed with minimal setup on any computer that possesses Python, which itself is quick to install [63]. More detail about the deployment of the algorithm is given in appendix A.

7.3 Limitations

This section introduces major limitations of the algorithm. Additional limitations and potential solutions are discussed in the following section 7.4.

7.3.1 Validation

As mentioned in the introduction of section 7.1, the ground truths used to assess the algorithm's performance were the same as those used for its development. This constitutes a significant bias in the metrics measured during the validation process and presented in section 7.1.

Furthermore, the dataset used to develop and test the pipeline consists of two CT scans. Hence, the dataset cannot be considered as representative of all possibly observable configurations of contacts in a CT image, or all ranges of CT artifacts strengths.

Finally, the ground truth annotations were generated by a non-expert, and with precision limited by the images' respective resolutions. Therefore, the actual errors could be slightly different from those presented in figure 7.4 and table 7.1.

All in all, section 7.1 showed that the algorithm is functional and produces reasonable outputs. But accounting for the three factors stated above, the exact results presented in that section cannot be interpreted as a reliable estimate of the algorithm's general performance.

Table 7.2: A summary of the key features of the existing tools and methods studied, as well as the method proposed in this thesis. Individual performances are not reflected. The blue dots could be removed with an enhanced contact generation module.

Method	Auto	NE	NC	EP	ICD	BM	It	Ar	Bd
Arnulfo et al. [13]	F			●			●		●
Meesters et al. [14]	F	●	●			●			
Qin et al. [15]	s				●	●			●
Granados et al. [16]	F								●
Benadi et al. [17]	F				●				
Lin et al. [18]	F				●	●	●	●	
Pantovic et al. [19]	F	*	*	*	*				●
Wong et al. [20]	F	●	●	●	●				
Chinara et al. [21]	s	●	●	●	●		●	●	
Proposed method	F	●	●	●		●	●	●	●

Acronym	Definition
Auto	Fully (F) or semi-automated (s)
NE	Requires number of electrodes
NC	Requires number of contacts per electrode
EP	Requires coordinates of electrode entry points
ICD	Requires inter-contact distance
BM	Requires brain mask
It	Robust to electrode crossings
Ar	Robust to strong CT artifacts and/or joint contacts after thresholding
Bd	Robust to electrode bending
*	During training only

7.3.2 Handling Shafts

In sEEG, a contact-less part of shaft is sometimes inserted into the brain as well, to allow an electrode to reach deeper structures. Figure 7.6 shows an example of that phenomenon. In theory, such parts do not constitute relevant information to the localization of the contacts, and they must be ignored. However, the current implementation of the algorithm allows generating centroids in the shaft, giving the impression that there are contacts to localize there as well. Hence, some contacts could be incorrectly predicted in that region of the electrode.

Parts of contact-less shafts are less common scenarios in sEEG that represent a corner case of the proposed algorithm. Among the other existing methods mentioned, only Chinara et al.’s [21] solution was able to handle those, since the method required the user to manually select the starting and ending positions of the contacts along the electrodes. None of the others methods showed robustness to such cases.



Figure 7.6: Presence of a contact-less shaft in Patient 2’s CT image. The electrode mask (transparent, gray) contains parts with (blue) and without (green) contacts.

7.4 Perspectives and Potential Improvements

This final section points out some of the algorithm’s minor limitations and drawbacks, and proposes solutions to them that could be implemented in future works. The code of the project is open source, such that enhancements provided to the algorithm can easily be implemented. Details about the organization of the source code can be found in appendix A.

7.4.1 Modeling of the Electrodes

In the post-processing module, curved electrodes are represented by parabolas. While this approach does account for electrode bending, the type of curvature that it can accurately fit is limited. For example, quadratic models are not well suited for electrodes with different changes of trajectory, as illustrated in figure 7.7.

Additionally, the implementation of parabola fitting to a set of points, detailed in appendix B, relies on the hypothesis of moderate bending to accurately fit a parabola to the centroids. Stronger curvature of the former results in increasingly imprecise approximations, and can therefore lead to potential errors in the directional parameters of the parabola. In practice, these errors are insignificant, but their very presence constitutes an issue for the algorithm’s robustness to stronger curvatures.

Overall, future works about this algorithm could benefit from an enhanced implementation of curved models, e.g. cubic curves, to allow covering more types of electrode bending.

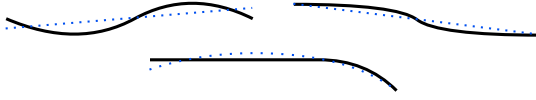


Figure 7.7: *A few particular types of electrode curvature (solid, black) poorly represented by parabolas (dashed, blue). The curvatures are exaggerated for clarity.*

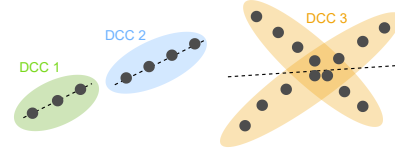


Figure 7.8: *Incorrect fit of $K = 3$ linear models (dashed) to centroids (points), due to a bad dilation radius R in the centroid extraction module.*

7.4.2 Detecting and Correcting Failures

The end of section 6.1.2 discussed the impact of the hyperparameter R , the dilation radius used to identify DCCs. In the case of a chosen R too small to connect the contacts of a given electrode, that section simply stated that the algorithm would fail, without entering details. The reason of this failure is explained here, followed by a proposition of solution. In the modeling module, it was explained that the graph search was optimized by forcing one-electrode DCCs to be handled in priority (see section 6.2.3). Hence, if a single electrode gets incorrectly split into two one-electrode DCCs, these will be handled in priority to fit linear models. With a constant number K of models to create, if one excess model is fit somewhere in the image, it does not leave enough models available to fit the rest of the electrodes, as illustrated in figure 7.8. In that figure, the electrode on the left has been incorrectly split into two one-electrode DCCs. Both are processed first by the modeling module, which fits one model to each. Finally, with $K = 3$ electrodes, only one model remains to fit the two electrodes in DCC 3. Thus, the algorithm fails to accurately represent the electrodes.

In a future implementation of the pipeline, the hyperparameter R could be found automatically by progressively dilating the EMSK, and stopping when the risk of disjoint contacts is sufficiently low, for example when the number of DCCs becomes lesser than the number K of electrodes. Figure 7.9 shows that the transition from $R = 1$ to $R = 2$ results in a drop of the number of DCCs, which becomes lesser than the number of electrodes. This can be assumed to be caused by the joining of the contacts in the EMSK. Hence, the value $R = 2$ would be used by the algorithm, as it maximizes the number of DCCs while minimizing the risk of disjoint contacts.

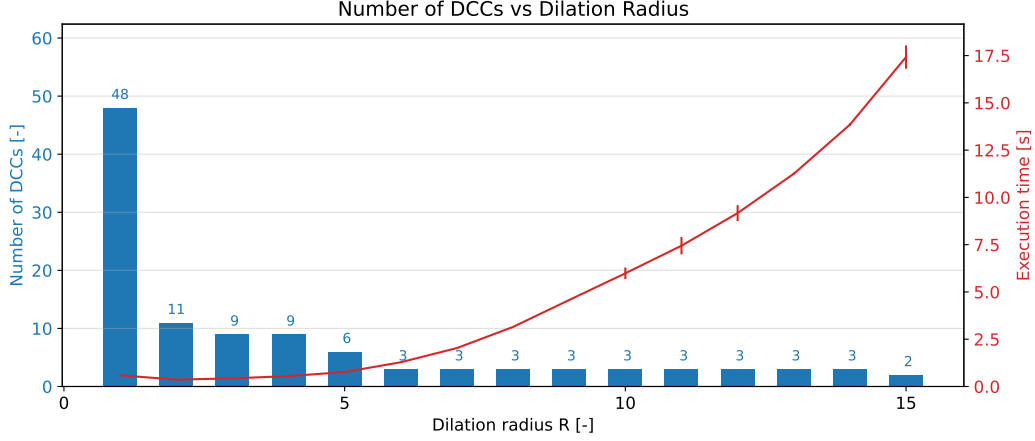


Figure 7.9: The effect of the dilation parameter R on the number of post-dilation connected components (DCCs) detected. The execution times were measured on five runs of the modeling module, with standard deviations indicated as vertical lines.

7.4.3 Revising the Contact Generation Module

Suppressing Shifts

Section 7.1.3 showed that the algorithm sometimes predicted contacts that were shifted with respect to the expected ones. The exact reason behind this phenomenon was not investigated, but can most likely be attributed to either or both of the following issues:

- The loss function used by the optimizer does not penalize shifts as expected.
- The optimizer in charge of minimizing the loss gets stuck in a local optimum.

In future works, the loss function and/or optimizing method could be revised to prevent shifted contact sequences from being selected as optimal.

Inputs

The algorithm requires numerous inputs compared to most other existing methods, as shown in table 7.2. The number of electrodes and brain mask constitute reasonable information to require; the former can be obtained visually in a matter of minutes with any CT image viewer, and the latter can be computed automatically by one of the several existing tools, such as FreeSurfer’s SynthStrip [58].

On the other hand, the contact generation module is responsible for the need of a specifically-formatted CSV file that contains both the entry points and number of contacts of all electrodes. These inputs are less convenient to obtain, and require approximately a dozen minutes of

manual labor from the neuroscientist or clinician. Furthermore, this task is sometimes made more complex by strong metal artifacts around intersections, where contacts can be difficult to distinguish.

In future works on the pipeline, the need for these two impractical inputs could be removed altogether by revising the contact generation module such that it automatically infers the optimal number of electrodes to generate along each model. For example, this could be achieved by computing sequences with increasing numbers of contacts along each model, and using a pre-defined metric to identify the best one.

7.4.4 Graphical User Interface (GUI)

The current implementation of the algorithm requires to be launched from a computer terminal, with the inputs specified in the arguments. This method is tedious and not practical for users who are not familiar with terminals. Developing a graphical user interface (GUI) to wrap the algorithm could strongly improve its ease of use. Through the GUI, the user could, for example, specify the inputs and parameters of the algorithm, view the results using a 3D viewer or a 2D slice viewer, and possibly make changes to the positions of the predicted contacts, to further improve accuracy. For this thesis, an early prototype of GUI was developed. It is displayed in figure 7.10. It allowed conveniently loading the input image and brain mask, and running the centroid extraction module. Centroids could then manually be added, moved, or deleted. Due to time constraints, development of a complete prototype could not be completed. In future works, implementing a fully-working GUI could significantly improve the user's experience with the algorithm.

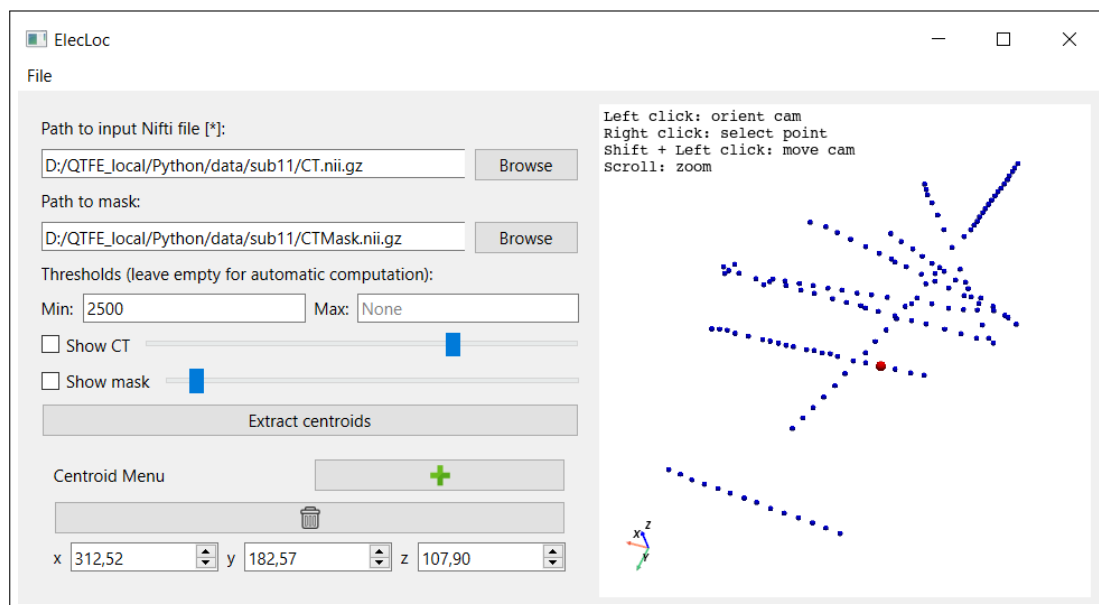


Figure 7.10: The early GUI prototype developed to wrap the algorithm. The left part allows to define the inputs and modify the results. The right part shows a 3D viewer displaying the output centroids of the extraction module.

Conclusion

This thesis has addressed a critical and time-consuming challenge in the field of epilepsy treatment and neuroscience research: the accurate and efficient localization of stereoelectroencephalography (sEEG) electrode contacts in post-operative computed tomography (CT) scans. Leveraging a combination of mathematical morphology, graph theory, and parabolic regression, a novel algorithm was developed to detect and label sEEG electrodes.

The proposed approach distinguishes itself from existing methods by its robustness to CT artifacts, electrode intersections, and electrode bending, which constitute common limitations in many existing tools. The algorithm starts by applying ultimate erosion to the thresholded and brain-masked CT image to obtain pointwise information about the electrodes despite complex trajectories, identifying intersections in the process. Then, the main axes of the electrodes are reconstructed using graph theory and an optimized search strategy. A final post-processing step accounts for electrode bending through parabolic regression, then retrieves optimal sequences of equidistant contacts along all electrodes.

The method was validated on clinical data from patients at Rigshospitalet, Copenhagen, demonstrating reliable performance in real-world settings. This pipeline significantly reduces the time and labor required to localize and label electrode contacts in a CT scan, compared to manual intervention, enhancing both the efficiency and reproducibility of the process.

In future works, the pipeline can benefit from several enhancements. These include more sophisticated modeling of curved electrodes, reduction of the inputs and hyperparameters required by the algorithm, and the development of a graphical user interface (GUI). A more thorough evaluation of the algorithm can help identify edge cases and ensure robust performance across a wider range of electrode configurations and artifact severities.

In conclusion, this work presents a meaningful step forward in the automation of sEEG electrode localization. By integrating image processing and algorithmic techniques, it contributes a robust, generalizable solution with the potential to significantly impact both clinical outcomes and research precision in the field of epilepsy.

References

- [1] J. S. Duncan, G. P. Winston, M. J. Koepp, and S. Ourselin, “Brain imaging in the assessment for epilepsy surgery,” *The Lancet Neurology*, vol. 15, no. 4, pp. 420–433, Apr. 2016, ISSN: 1474-4422. DOI: 10.1016/S1474-4422(15)00383-X. [Online]. Available: [http://dx.doi.org/10.1016/S1474-4422\(15\)00383-X](http://dx.doi.org/10.1016/S1474-4422(15)00383-X).
- [2] C. Baumgartner, J. P. Koren, M. Britto-Arias, L. Zoche, and S. Pirker, “Presurgical epilepsy evaluation and epilepsy surgery,” *F1000Research*, vol. 8, F1000-Faculty, 2019.
- [3] I. Bloch, H. Heijmans, and C. Ronse, “Mathematical Morphology,” *Handbook of Spatial Logics*, 2007. DOI: 10.1007/978-1-4020-5587-4_14.
- [4] J. Serra, “Introduction to mathematical morphology,” *Graphical Models /graphical Models and Image Processing /computer Vision, Graphics, and Image Processing*, 1986. DOI: 10.1016/0734-189X(86)90002-2.
- [5] R. J. Wilson, *Introduction to graph theory*, en, 4th ed. London, England: Prentice-Hall, 1996.
- [6] A. Bondy and U.S.R. Murty, *Graph Theory* (Graduate texts in mathematics), en, 1st ed. London, England: Springer, 2007.
- [7] J. Beisegel *et al.*, “The recognition problem of graph search trees,” *SIAM Journal on Discrete Mathematics*, vol. 35, no. 2, pp. 1418–1446, Jan. 2021, ISSN: 1095-7146. DOI: 10.1137/20M1313301. [Online]. Available: <http://dx.doi.org/10.1137/20M1313301>.
- [8] World Health Organization, *Epilepsy*, <https://www.who.int/news-room/fact-sheets/detail/epilepsy>, Last consulted 27/05/2025, 2024.
- [9] M. J. LaRiviere and R. E. Gross, “Stereotactic laser ablation for medically intractable epilepsy: The next generation of minimally invasive epilepsy surgery,” *Frontiers in Surgery*, vol. 3, Dec. 2016, ISSN: 2296-875X. DOI: 10.3389/fsurg.2016.00064. [Online]. Available: <http://dx.doi.org/10.3389/fsurg.2016.00064>.
- [10] L. Vaugier *et al.*, “The role of stereoelectroencephalography (seeg) in reevaluation of epilepsy surgery failures,” *Epilepsy and Behavior*, vol. 81, pp. 86–93, Apr. 2018, ISSN: 1525-5050. DOI: 10.1016/j.yebeh.2018.02.005. [Online]. Available: <http://dx.doi.org/10.1016/j.yebeh.2018.02.005>.
- [11] P. Jayakar *et al.*, “Diagnostic utility of invasive eeg for epilepsy surgery: Indications, modalities, and techniques,” *Epilepsia*, vol. 57, no. 11, pp. 1735–1747, 2016.
- [12] Lead-DBS, *About the MNI space(s)*, <https://www.lead-dbs.org/about-the-mni-spaces/>, Last consulted 30/04/2025, 2019.
- [13] G. Arnulfo, M. Narizzano, F. Cardinale, M. Fato, and J. M. Palva, “Automatic segmentation of deep intracerebral electrodes in computed tomography scans,” *BMC Bioinformatics*, 2015. DOI: 10.1186/s12859-015-0511-6.
- [14] S. Meesters *et al.*, “Automated identification of intracranial depth electrodes in computed tomography data,” *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, 2015. DOI: 10.1109/isbi.2015.7164034.

- [15] C. Qin *et al.*, “Automatic and precise localization and cortical labeling of subdural and depth intracranial electrodes,” *Front. Neuroinform.*, 2017. DOI: 10.3389/fninf.2017.00010.
- [16] A. Granados *et al.*, “Automatic segmentation of stereoelectroencephalography (seeg) electrodes post-implantation considering bending,” *International Journal of Computer Assisted Radiology and Surgery*, 2018. DOI: 10.1007/s11548-018-1740-8.
- [17] S. Benadi, I. Ollivier, and C. Essert, “Comparison of interactive and automatic segmentation of stereoelectroencephalography electrodes on computed tomography post-operative images: Preliminary results,” *Healthcare technology letters*, 2018. DOI: 10.1049/htl.2018.5070.
- [18] Z. Lin *et al.*, “A robust automated pipeline for localizing seeg electrode contacts,” *HBAI@IJCAI*, 2019. DOI: 10.1007/978-981-15-1398-5_3.
- [19] A. Pantovic, I. Ollivier, and C. Essert, “2d and 3d-unet for segmentation of seeg electrode contacts on post-operative ct scans,” *Medical Imaging*, 2022. DOI: 10.1117/12.2606538.
- [20] S. M. Wong, O. N. Arski, and G. M. Ibrahim, “An automated algorithm for stereoelectroencephalography electrode localization and labelling,” *Seizure*, 2024. DOI: 10.1016/j.seizure.2024.04.002.
- [21] C. Chinara *et al.*, “Automatic seeg contact localization and labeling using brainstorm,” Aug. 2024. DOI: 10.13140/RG.2.2.21742.11846.
- [22] International Society for Computed Tomography, *Half A Century In CT: How Computed Tomography Has Evolved*, <https://www.isct.org/computed-tomography-blog/2017/2/10/half-a-century-in-ct-how-computed-tomography-has-evolved>, Last consulted 25/04/2025, 2017.
- [23] L. Weinstein, “Electromagnetic Waves,” *Radio i svyaz’, Moscow*, 1988.
- [24] I. S. University, *Nature of Penetrating Radiation*, <https://www.nde-ed.org/Physics/X-Ray/nature.xhtml>, Last consulted 27/04/2025, 2021.
- [25] OpenMC, *Photon Physics*, https://docs.openmc.org/en/stable/methods/photon_physics.html, Last consulted 25/04/2025, 2023.
- [26] G. Kerckhofs, B. Macq, L. Dricot, and J. A. Lee, *LGBIO2050: Medical Imaging*, Course provided at UCLouvain, 2024.
- [27] Radiopaedia, *Beer-Lambert law*, <https://radiopaedia.org/articles/hounsfield-unit>, Last consulted 28/05/2025, 2025.
- [28] H. Hayashi *et al.*, “Generation of x-rays,” *Photon Counting Detectors for X-ray Imaging: Physics and Applications*, pp. 1–29, 2021.
- [29] J. I. Goldstein *et al.*, “X-rays,” *Scanning Electron Microscopy and X-Ray Microanalysis*, pp. 39–63, 2018.
- [30] O. Callis, *X-ray diagnostics and Computed tomography*, <https://www.slideserve.com/obelia/x-ray-diagnostics-and-computed-tomography>, Last consulted 28/05/2025, 2014.
- [31] O. Murray, *The anatomy of radiography*, <https://3d4medical.com/blog/the-anatomy-of-radiography>, Last consulted 28/05/2025, 2025.
- [32] L. Lechuga and G. A. Weidlich, “Cone Beam CT vs. Fan Beam CT: A Comparison of Image Quality and Dose Delivered Between Two Differing CT Imaging Modalities,” *Cureus*, 2016. DOI: 10.7759/cureus.778.
- [33] T. M. Buzug, “Computed Tomography,” *Springer eBooks*, 2011. DOI: 10.1007/978-3-540-74658-4_16.
- [34] Radiopaedia, *Hounsfield Unit*, <https://radiopaedia.org/articles/hounsfield-unit>, Last consulted 25/04/2025, 2015.
- [35] I. M. Laboratory, *Chapter 3. Coordinates and Transformations*, <https://motion.cs.illinois.edu/RoboticSystems/CoordinateTransformations.html>, Last consulted 26/04/2025, 2021.

- [36] F. E. Boas, D. Fleischmann, *et al.*, “CT artifacts: causes and reduction techniques,” *Imaging Med*, vol. 4, no. 2, pp. 229–240, 2012.
- [37] F. Boas and D. Fleischmann, “Evaluation of two iterative techniques for reducing metal artifacts in computed tomography,” *Radiology*, 2011. DOI: 10.1148/radiol.11101782.
- [38] J. Curot *et al.*, “Memory scrutinized through electrical brain stimulation: A review of 80 years of experiential phenomena,” *Neuroscience and Biobehavioral Reviews*, 2017. DOI: 10.1016/j.neubiorev.2017.04.018.
- [39] Z. Fu and U. Rutishauser, “Human single neuron recordings,” *Elsevier eBooks*, 2024. DOI: 10.1016/b978-0-12-820480-1.00002-4.
- [40] K. Iida and H. Otsubo, “Stereoelectroencephalography: Indication and efficacy,” *Neurologia medico-chirurgica*, 2017. DOI: 10.2176/nmc.ra.2017-0008.
- [41] D. medical, *Microdeep SEEG Electrodes*, <https://diximedus.com/products/microdeepseeg-electrodes/>, Last consulted 27/04/2025, 2023.
- [42] X. Fu *et al.*, “Carbon-based fiber materials as implantable depth neural electrodes,” *Frontiers in Neuroscience*, 2021. DOI: 10.3389/fnins.2021.771980.
- [43] A. Do, S. A. Menaker, J. L. Chan, and A. N. Mamelak, “Placement of stereotactic electroencephalography depth electrodes using the stealth autoguide robotic system: Technical methods and initial results,” *Operative Neurosurgery*, 2022. DOI: 10.1227/ons.0000000000000110.
- [44] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *Lecture Notes in Computer Science*, 2015. DOI: 10.1007/978-3-319-24574-4_28.
- [45] Ö. Çiçek, A. Abdulkadir, S. Lienkamp, T. Brox, and O. Ronneberger, “3d u-net: Learning dense volumetric segmentation from sparse annotation,” *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2016. DOI: 10.1007/978-3-319-46723-8_49.
- [46] D. T. Hub, *Using binary to create on/off pictures*, <https://www.digitaltechnologieshub.edu.au/teach-and-assess/classroom-resources/lesson-ideas/using-binary-to-create-on-off-pictures/>, Last consulted 29/05/2025, 2022.
- [47] X. Benavent, E. Durá, F. Vegara, and J. Domingo, “Mathematical Morphology for Color Images: An Image-Dependent Approach,” 2012. DOI: 10.1155/2012/678326.
- [48] R. M. Gibson, A. Ahmadiania, S. G. McMeekin, N. C. Strang, and G. Morison, “A reconfigurable real-time morphological system for augmented vision,” *EURASIP Journal on Advances in Signal Processing*, vol. 2013, no. 1, Aug. 2013, ISSN: 1687-6180. DOI: 10.1186/1687-6180-2013-134. [Online]. Available: <http://dx.doi.org/10.1186/1687-6180-2013-134>.
- [49] L. Vincent, “Morphological grayscale reconstruction in image analysis: applications and efficient algorithms,” *IEEE Transactions on Image Processing*, 1993. DOI: 10.1109/83.217222.
- [50] L. Euler, “The seven bridges of königsberg,” *The world of mathematics*, vol. 1, no. 1, pp. 573–580, 1956.
- [51] MathSpace, *Kruskal’s algorithm*, <https://mathspace.co/textbooks/syllabuses/Syllabus-320/topics/Topic-5916/subtopics/Subtopic-119166/?activeTab=theory>, Last consulted 23/05/2025, 2022.
- [52] D. G. Corneil and R. M. Krueger, “A unified view of graph searching,” *SIAM Journal on Discrete Mathematics*, vol. 22, no. 4, pp. 1259–1276, Jan. 2008, ISSN: 1095-7146. DOI: 10.1137/050623498. [Online]. Available: <http://dx.doi.org/10.1137/050623498>.
- [53] R. Graham, R. Perriss, and A. Scarsbrook, “Dicom demystified: A review of digital file formats and their use in radiological practice,” *Clinical Radiology*, vol. 60, no. 11, pp. 1133–

- 1140, 2005, ISSN: 0009-9260. DOI: 10.1016/j.crad.2005.07.003. [Online]. Available: <http://dx.doi.org/10.1016/j.crad.2005.07.003>.
- [54] *Pydicom*, <https://pydicom.github.io/>, Last consulted 17/04/2025, 2018.
 - [55] National Institute of Health, *NIfTI-2 Data Format*, <https://nifti.nimh.nih.gov/nifti-2/>, Last consulted 27/05/2025, 2020.
 - [56] M. e. a. Brett, *Nipy/nibabel: 5.3.1*, <https://zenodo.org/doi/10.5281/zenodo.591597>, 2024. DOI: 10.5281/ZENODO.591597.
 - [57] X. Li, P. S. Morgan, J. Ashburner, J. Smith, and C. Rorden, “The first step for neuroimaging data analysis: Dicom to nifti conversion,” *Journal of Neuroscience Methods*, 2016. DOI: 10.1016/j.jneumeth.2016.03.001.
 - [58] A. Hoopes, J. S. Mora, A. V. Dalca, B. Fischl, and M. Hoffmann, “Synthstrip: Skull-stripping for any brain image,” *NeuroImage*, 2022. DOI: 10.1016/j.neuroimage.2022.119474.
 - [59] Y.-C. Chen, “A tutorial on kernel density estimation and recent advances,” en, vol. 1, no. 1, pp. 161–187, 2017.
 - [60] S. J. Miller, “The method of least squares,” *Mathematics Department Brown University*, vol. 8, no. 1, pp. 5–11, 2006.
 - [61] D. G. McVitie and L. B. Wilson, “The stable marriage problem,” *Communications of the ACM*, vol. 14, no. 7, pp. 486–490, Jul. 1971, ISSN: 1557-7317. DOI: 10.1145/362619.362631. [Online]. Available: <http://dx.doi.org/10.1145/362619.362631>.
 - [62] B. Guenter and R. Parent, “Computing the arc length of parametric curves,” *IEEE Computer Graphics and Applications*, vol. 10, no. 3, pp. 72–78, 1990, ISSN: 0272-1716. DOI: 10.1109/38.55155. [Online]. Available: <http://dx.doi.org/10.1109/38.55155>.
 - [63] Python Software Foundation, *Python*, <https://www.python.org/>, Last consulted 1/06/2025, 2001.

Appendices

Appendix A

Code and Repository Overview

The code developed for this thesis can be found in the following repository: https://github.com/Quent-DL/MasterThesis_ElectroLoc. It is entirely open source, and open to improvements and future work.

A.1 General Information about the Project

The `README.md` file in the repository contains all relevant information regarding the use or development of the algorithm, particularly:

- a quick installation guide;
- details about simply executing the algorithm with default hyperparameters, or customized ones;
- an explanation of the expected input files, and the ones returned;
- a presentation of the project’s organization.

For more detail about the repository or the algorithm, the reader is kindly redirected to the `README.md` file.

A.1.1 Inputs of the Algorithm

As discussed in section 5.2.1, the CT volume and brain mask processed must be in NIfTI format to be exploitable by the algorithm. More specifically, they must be compressed, in the `.nii.gz` format. On the other hand, the electrode information file is expected to be a CSV with the format described in table A.1.

Table A.1: *The format expected from the input CSV file that holds information about the electrodes.*

<code>nb_contacts</code>	<code>vox_x</code>	<code>vox_y</code>	<code>vox_z</code>
10	310	262	34
12	315	205	47
...			

In this table, each row holds information about one electrode. The field `nb_contacts` contains the number of contacts to generate along each electrode (see section 6.3.2). The three fields

`vox_x`, `vox_y`, `vox_z` represent the entry point coordinates of the electrodes, expressed in voxel space.

The package also offers the possibility to provide a ground truth with the CT image to process, to assess the performance of the algorithm on this image. This ground truth must also be a CSV file, whose fields are described in table A.2.

Table A.2: *The format expected from the optional ground truth CSV file provided to the algorithm.*

<code>vox_x</code>	<code>vox_y</code>	<code>vox_z</code>	<code>electrode_id</code>	<code>c_id</code>
310	262	34	0	0
315	261	36	0	1
...				

In this table, each row matches a single contact. The coordinates `vox_x`, `vox_y`, `vox_z` are the center of the contact, in voxel space. The field `electrode_id` is the equivalent of the EID used throughout this paper. Finally, `c_id` is the equivalent of the PID, *i.e.* the position of the contact onto its electrode.

A.1.2 Outputs of the Algorithm

The algorithm proposes two types of output files: a CSV with a summary of the contacts generated, or a contact mask. The latter is returned as a binary `.nii.gz` file in which the foreground is constituted of the center of all contacts predicted, as well as some voxels around, for clarity.

If asked by the user, the CSV file with the summary of the contacts generated is returned in the format presented in table A.3.

Table A.3: *The format of the CSV file returned by the algorithm, with information about all the contacts generated.*

<code>vox_x</code>	<code>vox_y</code>	<code>vox_z</code>	<code>world_x</code>	<code>world_y</code>	<code>world_z</code>	<code>electrode_id</code>	<code>c_id</code>
294.048	262.64	92.612	-17.835	1.696	138.313	0	0
301.557	262.424	92.485	-21.355	1.595	138.234	0	1
...							

As in the ground truth, the fields `vox_[x, y, z]` represent the centers in voxel space of the contacts. The values `world_[x, y, z]` hold the same values, but expressed in physical space. Both coordinate spaces are given for convenience. The fields `electrode_id` represent the same information `c_id` as those in the ground truth.

Appendix B

Time-Parametrized Parabolic Curves

This appendix delves into the mathematical definitions and computations used to build and fit 3-dimensional parabolas to a set of points. Its content is used in section 6.3.1 of the thesis.

B.1 Definition of 3-Dimensional Parabolas

In 3-dimensional geometry, a parabola is a particular type of time-parametrized curve that lies on a specific plane of $\mathbb{R}^3$, and resembles a 2D parabola within that plane. Such a curve is defined as

$$\begin{aligned} \mathbf{x}_p(t) &= \begin{bmatrix} x_p(t) \\ y_p(t) \\ z_p(t) \end{bmatrix} = \begin{bmatrix} a \\ d \\ g \end{bmatrix} t^2 + \begin{bmatrix} b \\ e \\ h \end{bmatrix} t + \begin{bmatrix} c \\ f \\ i \end{bmatrix} \\ &= \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} t^2 \\ t \\ 1 \end{bmatrix} =: \mathbf{A} \mathbf{t} \quad \forall t \in [t_a; t_b] \end{aligned} \tag{B.1}$$

where $a, \dots, i \in \mathbb{R}$ are the directional parameters of the parabola p , which can be compactly described as the matrix $\mathbf{A} \in \mathbb{R}^{3 \times 3}$. The value $t \in \mathbb{R}$ is the free variable that parametrizes the points along the curve. It is bounded by t_a and $t_b \in \mathbb{R}$, to represent only a portion of parabola delimited by two points instead of a full one. The scalar time parameter t must not be mistaken with the vector $\mathbf{t} := (t^2, t, 1) \in \mathbb{R}^3$. Figure B.1 illustrates an example of parabolic curve and shows the impact of the parameters on its shape.

To find the projection of a point $\mathbf{x}$ onto the parabola p , there exists no trivial, static formula that depends on $\mathbf{A}$. Instead, the projection must be computed through an explicit optimization problem:

$$\begin{aligned} \text{proj_time}_p(\mathbf{x}) &= t^* := \arg \min_{t \in \mathbb{R}} \|\mathbf{x} - \mathbf{x}_p(t)\| = \arg \min_{t \in \mathbb{R}} \|\mathbf{x} - \mathbf{A} \mathbf{t}\| \\ \text{proj}_p(\mathbf{x}) &:= \mathbf{x}_p(t^*) = \mathbf{A} \mathbf{t}^* \end{aligned} \tag{B.2}$$

However, to compute the projection of a point $\mathbf{x}_0$ whose projection time $t_0 \in \mathbb{R}$ is hypothetically known, the resulting position on the parabola p can be obtained through the straightforward

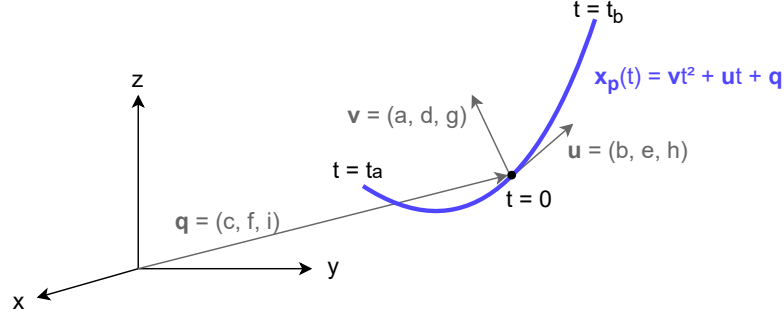


Figure B.1: An example of parabolic curve, defined by three directional vectors $\mathbf{q}$, $\mathbf{u}$, and $\mathbf{v}$. The ends of the parabola are restricted at times t_a and t_b .

application of formula B.1, then re-expressed as follows:

$$\begin{aligned}
 \mathbf{x}(t_0) &= \mathbf{A}\mathbf{t}_0 = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} t_0^2 \\ t_0 \\ 1 \end{bmatrix} \\
 \Leftrightarrow \begin{bmatrix} x(t_0) \\ y(t_0) \\ z(t_0) \end{bmatrix} &= \begin{bmatrix} at_0^2 + bt_0 + c \\ dt_0^2 + et_0 + f \\ gt_0^2 + ht_0 + i \end{bmatrix} \\
 &= \begin{bmatrix} t_0^2 & t_0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & t_0^2 & t_0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_0^2 & t_0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ \vdots \\ h \\ i \end{bmatrix} \\
 &=: \mathbf{T}_0 \mathbf{a}.
 \end{aligned} \tag{B.3}$$

This reformulation of B.1 switches the format of the parameters involved; time gets re-expressed as a matrix $\mathbf{T}_0 \in \mathbb{R}^{3 \times 9}$, and the parabola's directional parameters as a vector $\mathbf{a} \in \mathbb{R}^9$.

B.2 Fitting a Parabola to a Set of Points

This section details the methodology used to fit a parabola to a set of n distinct points defined as $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^3$. The goal is to retrieve the matrix $\mathbf{A}$ that best fits these points.

Given n distinct points ($n > 3$) defined as $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^3$ whose time parameters $t_1, \dots, t_n \in \mathbb{R}$ are hypothetically known along an unknown parabola, the parameters of the parabola that perfectly fits all the points, if it existed, could be obtained by solving the following linear¹ system

¹The system is indeed linear despite the presence of the squared values t_i^2 , because these are constants. The actual variables of the system are in the array $\mathbf{a}$.

for $\mathbf{a}$:

$$\begin{bmatrix} x(t_1) \\ y(t_1) \\ z(t_1) \\ x(t_2) \\ \vdots \\ z(t_{n-1}) \\ x(t_n) \\ y(t_n) \\ z(t_n) \end{bmatrix} = \begin{bmatrix} t_1^2 & t_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & t_1^2 & t_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_1^2 & t_1 & 1 \\ t_2^2 & t_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & t_{n-1}^2 & t_{n-1} & 1 \\ t_n^2 & t_n & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & t_n^2 & t_n & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_n^2 & t_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ \vdots \\ h \\ i \end{bmatrix} \quad (\text{B.4})$$

$$\Leftrightarrow \mathbf{x} = \mathbf{T}\mathbf{a},$$

where $\mathbf{x} \in \mathbb{R}^{3n}$ and $\mathbf{T} \in \mathbb{R}^{3n \times 9}$. But the hypotheses stated above— $n > 3$ and distinct points—render the system unsolvable. However, the best approximation for $\mathbf{a}$ can be obtained through the least squares method [60]:

$$\mathbf{a} = \arg \min_{\mathbf{a}' \in \mathbb{R}^9} \|\mathbf{x} - \mathbf{T}\mathbf{a}'\|^2 \quad (\text{B.5})$$

Hence, the least squares method allows regressing parabolic models to fit a given set of points. At least, this would be the case if the projection times $t_1, \dots, t_n$ of the points were known, which they are not. But as seen in formula B.2, computing these projection times requires the parabola's directional parameters. Therefore, trying to fit a time-parametrized parabola using only a set of points is a chicken-and-egg problem.

To fix this issue, the time parameters $t_1, \dots, t_n$ are instead estimated from linear regression of the points. Suppose a parabola p is defined as $\mathbf{x}_p(t) := \mathbf{v}t^2 + \mathbf{u}t + \mathbf{q}$, with $\mathbf{v}, \mathbf{u}, \mathbf{q} \in \mathbb{R}^3$ defined as the equivalent of the vectors in formula B.1. As $\mathbf{v}$ tends to the null vector, *i.e.* as the parabola becomes straighter, the projection time of a given point onto the parabola tends to the projection time of that same point onto the parametrized line l , defined by $\mathbf{x}_l(t) := \mathbf{u}t + \mathbf{q}$. Since the projection time of a point along a line is computable explicitly (see formula 6.3), it can be used as an estimate of the projection time along the parabola. The straighter the parabola, the better the estimate. Figure B.2 illustrates this principle. Therefore, this method is used to estimate the projection times of the centroids onto the parabola to fit, following the assumption mentioned earlier about reasonable curvature of the electrodes.

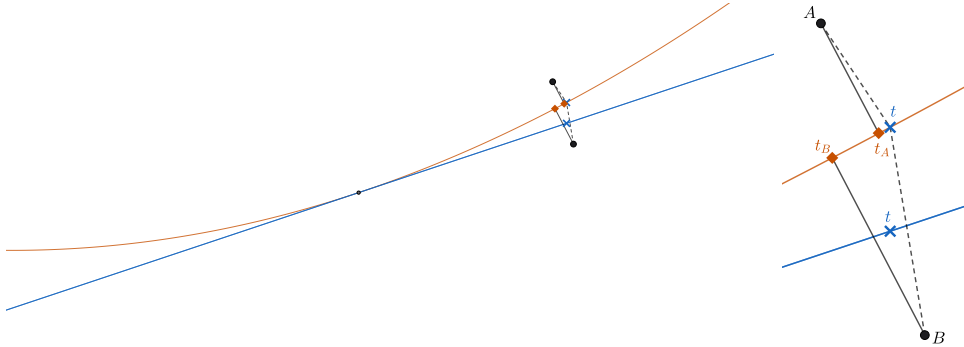


Figure B.2: *The assumption about the projection time of the samples used to fit a parabola. Time t is the exact projection time of A and B onto the blue line. While fitting the red parabola, the exact projection times t_A and t_B (solid lines) of points A and B respectively are approximated by t (dashed line).*

Additionally, the temporal bounds of the parabola obtained are defined as the lowest and highest projection times of the points $\mathbf{x}_1, \dots, \mathbf{x}_n$ along that parabola:

$$t_a := \min_{\mathbf{x}' \in \{\mathbf{x}_1, \dots, \mathbf{x}_n\}} \text{proj_time}_p(\mathbf{x}'), \quad t_b := \max_{\mathbf{x}' \in \{\mathbf{x}_1, \dots, \mathbf{x}_n\}} \text{proj_time}_p(\mathbf{x}'). \quad (\text{B.6})$$

B.3 Ignoring Outliers

If the set of points used to fit a parabola contains an outlier that lies far from the general parabolic shape, it can significantly affect the result. Since the least squares method minimizes the sum of squared errors, a distant outlier contributes a disproportionately large error. As a result, the method will adjust the parabola to reduce the impact of that outlier, leading to overfitting.

To avoid that, the classic least squares method is replaced by a weighted version. All points are given a weight that represents the proportional contribution of that point's error to the total loss, minimized by the least squares method. It is used to mitigate the impact of outliers on the loss function.

In the context of the thesis, where parabolic models are fit from a set of centroids on a line segment model, the weights are defined using formula 6.6. Hence, centroids that lie far away from that line segment (*i.e.* outliers) tend to have less impact during the fitting of the parabola.

The weights of all points $\mathbf{x}_1, \dots, \mathbf{x}_n$ are computed and gathered into the diagonal matrix

$$\mathbf{W} := \text{diag}(w_1, w_1, w_1, w_2, \dots, w_{n-1}, w_n, w_n, w_n) \in \mathbb{R}^{3n \times 3n}, \quad (\text{B.7})$$

that is used to weight the individual equations of the least squares problem as follows:

$$\mathbf{a} = \arg \min_{\mathbf{a}' \in \mathbb{R}^9} \|\mathbf{W}(\mathbf{x} - \mathbf{T}\mathbf{a}')\|^2 \quad (\text{B.8})$$

As a result, though outliers produce inconveniently big values in the global error vector $\mathbf{x} - \mathbf{T}\mathbf{a}'$, the weight matrix $\mathbf{W}$ causes the least squares method to ignore the rows associated with these outliers.

B.4 Summary of the Process

This section summarizes how a parabola is fit to a set of centroids $\mathbf{c}_1, \dots, \mathbf{c}_n \in \mathbb{R}^3$ all associated with the same line segment model, typically returned by the modeling module detailed in section 6.2.3.

1. The projection times of the centroids onto the line model are retrieved using formula 6.3.
2. These values are used as estimates of the projection times of those same centroids onto the parabolic model. Put differently, these values are used to build the matrix $\mathbf{T}$ as defined in formula B.4.
3. The vector $\mathbf{x}$ is built from the centroids' coordinates as defined in formula B.4.
4. The centroids' weights $w_1, \dots, w_n$ are computed using formula 6.6 and gathered into the diagonal weight matrix $\mathbf{W}$ (see formula B.7) to reduce the impact of outliers on the resulting parabola.

5. The weighted least squares method defined in formula B.8 is used to fit a parabola to the weighted centroids and retrieve the optimal directional parameters $\mathbf{a}$.

This process is then repeated for all line models and their associated centroids.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl