



People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University Aboubakr Belkaid Tlemcen
Faculty of Technology

Department of Electrical and Electronics
Engineering
Speciality: Automation
Option: Automation and Industrial Computing

Master's Thesis

Automated Design Optimization of TMS Coils
Using Machine Learning

Presented by:

AOUAM Mohammed & KAMNI Samaa Edounia

Defended on 21/06/2026 in front of the jury

<i>PRESIDENT :</i>	Mrs. BENREZZAK Nawel	MAA, Tlemcen University
<i>EXAMINATOR :</i>	Mr. BENAHBIB Mohammed Choukri	Pr, Tlemcen University
<i>EXAMINATOR :</i>	Mr. GORINE Ali	MCB, Tlemcen University
<i>SUPERVISOR :</i>	Mrs. HANDOUZI Wahida	MCA, Tlemcen University
<i>GUEST :</i>	Mr. ABDELALI Youssouf	MA, CHU Tlemcen

Dedication

I dedicate this modest work:

I dedicate this modest work:

- To my dear parents, for their love, their invaluable support, their sacrifices, and their encouragement throughout my studies. May God protect them and grant them health and happiness.
- To my family and loved ones, for their presence, their patience, and their valuable advice.
- To my teachers and supervisors, for their guidance, their availability, and the knowledge they have shared with me.
- To all my friends and classmates, who shared with me the moments of hard work, difficulty, and success.
- Finally, to all the people who contributed, directly or indirectly, to the completion of this end of studies project.

KAMNI Samaa Edounia

I dedicate this work to:

- My Beloved Parents: For their endless love, constant prayers, and the incredible sacrifices they made to see me reach this milestone. This achievement belongs to you just as much as it belongs to me.
- My Partners in the Field: To my dear friends and classmates, especially those who shared the same theme, the long nights of coding, debugging the 3D CNN, and pushing through the challenges of this project. Your constant motivation and shared struggles made this journey unforgettable.
- My Erasmus Circle in Romania: To all the fantastic local and international friends I met during my mobility time especially those from the Shakespeare Festival. Thank you for the laughs, the late-night talks, and for giving me a home away from home.

AOUAM Mohammed

Acknowledgments

First of all, we thank God Almighty for giving us courage, patience, and strength throughout all these years of study.

We would like to sincerely thank our supervisor, **HANDOUZI WAHIDA**, for her guidance, her availability, her valuable advice, and her support throughout the completion of this work.

We would also like to express our sincere gratitude to the members of the jury : Mrs BENREZZAK Nawel, Mr. BENAHBIB Mohammed Choukri and Mr. GORINE Ali for the honor they have given us by accepting to evaluate this work and for their valuable remarks and suggestions.

We also thank all our teachers for the knowledge, training, and encouragement they gave us during our university studies.

@Kamni, My thanks also go to the Tlemcen Hospital, and in particular to the neurology department, and Mr ABDELALLI Youssouf for welcoming us and allowing us to carry out this project in good conditions.

Finally, we would like to express our deep gratitude to all the people who contributed, directly or indirectly, to the completion of this end of studies project.

Abstract

Transcranial Magnetic Stimulation (TMS) is a non-invasive brain stimulation technique used to treat neurological disorders including treatment-resistant depression, obsessive-compulsive disorder, and post-stroke motor impairment. Its therapeutic effectiveness depends critically on the precise positioning of a magnetic coil on the patient’s scalp.

Knowing exactly where the stimulation electric field reaches inside the brain – for any given coil position – requires solving Maxwell’s equations over a detailed head mesh. The standard tool, Finite Element Method (FEM) simulation via SimNIBS, takes approximately three minutes per position on a normal workstation. This delay makes interactive, real-time positioning supervision impossible in a clinical setting.

This project addresses the problem from the perspective of Automation and Industrial Computing. A surrogate modelling approach directly analogous to the Digital Twin concept used in Industry 4.0 is applied. A 3D Convolutional Neural Network (TMS_CNN_3D) is trained on 50 automatically generated FEM simulations. It learns the mapping from a 16-scalar coil configuration vector to a complete $64 \times 64 \times 64$ electric field volume. A real-time Human-Machine Interface (HMI) integrates the trained model, allowing an operator to explore coil positions interactively.

The CNN achieves a Pearson correlation of $R = 0.887$ and a Mean Absolute Error of 2.19% on a normalised field scale on the validation set (10 simulations). Inference runs in approximately 150 ms on a standard Intel Core i5 CPU – a factor of 1,200 faster than FEM – meeting the 200 ms real-time supervision target. No GPU is required.

Keywords: Transcranial Magnetic Stimulation, System Identification, Surrogate Model, 3D Convolutional Neural Network, Digital Twin, Real-Time Supervision, Human-Machine Interface, Automation.

Résumé

La stimulation magnétique transcrânienne (TMS) est une technique non invasive de stimulation cérébrale utilisée pour traiter plusieurs troubles neurologiques et psychiatriques, notamment la dépression résistante aux traitements, les troubles obsessionnels compulsifs et les déficits moteurs post-AVC. En pratique clinique, son efficacité dépend fortement de la précision du positionnement de la bobine magnétique sur le cuir chevelu, car de légères variations peuvent modifier significativement le champ électrique induit dans le cerveau.

Pour estimer la distribution de ce champ électrique pour une position donnée de la bobine, il est nécessaire de résoudre les équations de Maxwell sur un modèle réaliste de la tête. Cette opération est généralement réalisée à l'aide de la méthode des éléments finis (FEM), notamment via le logiciel SimNIBS. Bien que précise, cette approche est coûteuse en temps de calcul, nécessitant environ trois minutes par simulation sur une station de travail classique, ce qui limite son utilisation en temps réel en milieu clinique.

Dans ce projet, nous proposons une approche inspirée de l'Automatique et de l'Informatique Industrielle, basée sur un modèle de substitution analogue au concept de jumeau numérique (Digital Twin) utilisé dans l'Industrie 4.0. Un réseau de neurones convolutionnel 3D, appelé `TMS_CNN_3D`, est entraîné à partir de 50 simulations FEM générées automatiquement. Ce modèle apprend à prédire un volume complet de champ électrique de taille $64 \times 64 \times 64$ à partir d'un vecteur de 16 paramètres décrivant la configuration de la bobine. Il est ensuite intégré dans une interface homme-machine (IHM) permettant une exploration interactive et en temps réel des positions de stimulation.

Les résultats obtenus montrent une corrélation de Pearson de $R = 0.887$ et une erreur absolue moyenne de 2.19% sur un ensemble de validation composé de 10 simulations. Le temps d'inférence est d'environ 150 ms sur un processeur Intel Core i5 standard, soit une accélération d'un facteur 1 200 par rapport à la FEM, répondant ainsi aux contraintes du temps réel sans recours à un GPU.

Mots-clés : Stimulation Magnétique Transcrânienne, Identification de systèmes, Modèle de substitution, Réseau de neurones convolutionnel 3D, Jumeau numérique, Temps réel, Interface Homme-Machine, Automatique.

List of Acronyms

AIC	Automation and Industrial Computing
API	Application Programming Interface
BN	Batch Normalisation
CNN	Convolutional Neural Network
ENCODER OC	Encoder Overcomplete
CPU	Central Processing Unit
CSF	Cerebrospinal Fluid
DLPFC	Dorsolateral Prefrontal Cortex
DNN	Deep Neural Network
DOPA	Digital Optimisation and Process Automation
FDA	Food and Drug Administration (USA)
FEM	Finite Element Method
GPU	Graphics Processing Unit
HMI	Human-Machine Interface
MAE	Mean Absolute Error
ML	Machine Learning
MLP	Multi-Layer Perceptron
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NLARX	NonLinear AutoRegressive with eXogenous input
OCD	Obsessive-Compulsive Disorder
PFE	Projet de Fin d'Études (End-of-Studies Project)
ReLU	Rectified Linear Unit
ROI	Region of Interest
SCADA	Supervisory Control and Data Acquisition
SimNIBS	Software for Non-Invasive Brain Stimulation
TMS	Transcranial Magnetic Stimulation
UML	Unified Modelling Language

Contents

Dedication	2
Acknowledgments	3
Abstract	4
Résumé	5
List of Acronyms	6
General Introduction	11
0.1 Problem Statement and motivation	12
0.2 Research Objectives and methodology	12
0.3 Thesis Organisation	13
1 Control Engineering Framework for Transcranial Magnetic Stimulation (TMS)	14
1.1 Physical Characterisation of the TMS Actuator	15
1.1.1 Electromagnetic Induction Principles and Neuron Activation Mechanism	15
1.1.2 Coil Geometry and Spatial Selectivity of the Actuator Output . .	16
1.2 Input Output Modelling of the TMS System	17
1.2.1 System Input: Coil Configuration as a State Vector	17
1.2.2 System Output: Volumetric Electric Field Distribution	18
1.2.3 The Physical Transfer Operator: Forward and Inverse Problems .	18
1.3 Computational Constraints: The FEM Bottleneck	19
1.4 State of the Art: Surrogate Models and Deep Learning in Neurostimulation	19
1.4.1 Surrogate Modelling in Control Engineering	19
1.4.2 Deep Learning Approaches to E Field Prediction	19
1.4.3 Connection to Control and Industrial Computing	20
1.5 Gap Analysis and Project Objectives	22
1.5.1 Identified Gaps in the Literature	22
1.5.2 Primary Objective and Contributions	22
1.6 Conclusion	22
2 System Identification and Data Generation	23
2.1 Surrogate Modelling Strategy	23
2.2 Reference Modelling by Finite Elements	24
2.2.1 Spatial Discretisation and Tissue Conductivities	24

2.2.2	The SimNIBS 4.5 Simulation Environment	24
2.3	Dataset Generation Protocol	25
2.4	Signal Preprocessing and Normalisation	26
2.5	Dataset Overview	27
2.6	Architecture of the 3D Neural Network	28
2.6.1	MLP Encoder overcomplete: Processing the Geometric Input . . .	29
2.6.2	3D Projection Layer	29
2.6.3	3D Convolutional Decoder: Progressive Upsampling	30
2.7	Non Linear Identification: Cost Function and CPU Optimisation	31
2.7.1	Cost Function: MSE on 3D Volumes	31
2.7.2	Optimiser and Learning Rate Scheduling	31
2.7.3	Training Hyperparameters	32
2.7.4	Data Augmentation	32
2.8	Conclusion	32
3	Implementation of the Real Time Supervision System and Human Machine Interface	34
3.0.1	Learning Curves and Convergence Analysis	34
3.1	Performance Analysis of the Identified Model	35
3.1.1	Global Performance Metrics	36
3.1.2	Performance by Simulation	36
3.1.3	Comparative Visualisations: Ground Truth vs Prediction	36
3.1.4	Positioning Relative to the Literature	39
3.2	Validation of Computational Complexity Reduction	40
3.2.1	Inference Time Measurement	40
3.2.2	Speed Gain Compared to SimNIBS FEM	40
3.2.3	Characteristics of the Best Saved Model	40
3.3	Architecture of the Real Time Supervision System	41
3.3.1	Replacing FEM with the Surrogate Model	41
3.3.2	System Data Flow	41
3.3.3	Integration of the Trained Model	42
3.4	Human Machine Interface Design	42
3.4.1	Design Objectives and Industrial Supervision Principles	42
3.4.2	Coil Position Controls: Parametric Sliders	42
3.4.3	Multi Plane Electric Field Visualisation	43
3.4.4	Software Implementation and Event Driven Architecture	43
3.5	Results and Discussion: Latency Analysis and Real Time Compliance . .	46
3.5.1	The 200 ms Real Time Constraint	46
3.5.2	Comparison with the State of the Art	46
3.5.3	Impact of Positioning Error on Field Estimation	47
3.6	Perspectives: Toward Closed Loop Control and Automatic Optimisation	47
3.6.1	Current Limitation: Open Loop Operation	47
3.6.2	Gradient Based Optimisation Through the Differentiable Network	47
3.6.3	Bayesian Optimisation	47
3.6.4	Roadmap Toward a Closed Loop TMS Positioning System	47
3.7	Conclusion	48
	General conclusion	49

List of Figures

1.1	Input and output data used for training.	16
1.2	(a) The TMs used in this project, (b) The principle of TMS coil	17
1.3	Block Diagram - TMS system assisted by CNN	21
2.1	Head Modeling Approaches for TMS Simulations	25
2.2	Preprocessing pipeline illustration: (a) irregular tetrahedral mesh output from SimNIBS, (b) brain region extraction, (c) interpolated 64^3 voxel grid, (d) normalised field volume.	27
2.3	network MLP encoder overcomplete	29
2.4	The architecture of the 3DCNN decoder	30
3.1	MSE loss curves (training and validation) over 150 epochs. Left: linear scale. Right: logarithmic scale	35
3.2	Sample 3 (lateral, MAE=0.0229, $R = 0.8646$): FEM vs CNN vs error map	37
3.3	Sample 18: intermediate position. Slight underestimation of the field gradient at the tissue interface	38
3.4	Sample 34: best performance ($R = 0.9076$). Top pole configuration is the best represented in training data	39
3.5	(a) Visualization of Electrode Currents, Electric Field Distribution, and Variability in the Brain, (b) Statistical Parametric Maps Showing Significant Brain Activation Differences	43
3.6	Graphical user interface for real-time electric field visualization and TMS coil positioning supervision	45

List of Tables

1.1	Composition of the 4×4 homogeneous transformation matrix (CNN input).	18
1.2	TMS modelled as an input output system within the control engineering frame	18
1.3	Comparison of computation times across clinical scenarios: FEM versus the 3D CNN surrogate.	19
1.4	Comparative summary of deep learning approaches to TMS E field prediction.	20
1.5	Summary of the three project contributions and the results obtained. . .	22
2.1	Electrical conductivities of head tissues in the full SimNIBS model. In our simplified spherical model, only scalp and grey matter are used.	24
2.2	Dataset composition after generation (raw SimNIBS outputs)	25
2.3	Preprocessing pipeline: irregular mesh 64^3 voxel grid	26
2.4	Parameters of the interpolation ROI grid.	27
2.5	Complete identification dataset summary	28
2.6	Parameters of the interpolation ROI grid.	28
2.7	Detailed architecture of the 3D CNN TMS_CNN_3D (2,308,673 total parameters)	30
2.8	Justification of the architectural choices of the 3D CNN	31
2.9	Training hyperparameters of the TMS 3D CNN	32
3.1	Evolution of MSE losses during training (extract from metadata.json) . .	35
3.2	Performance metrics on the validation set (10 simulations, metadata.json)	36
3.3	Detailed performance on the three illustrated simulations	36
3.4	Positioning of performance relative to reference studies in the literature .	39
3.5	Breakdown of inference time by step (Intel i5 7th gen, CPU only)	40
3.6	Comparison of computation times: FEM vs 3D CNN	40
3.7	Characteristics of the best model (best_model.pth)	41
3.8	Data flow of the supervision system	42
3.9	Features of the developed HMI interface	43
3.10	Breakdown of the measured latency budget (Intel i5 7th gen CPU, no GPU).	46
3.11	Roadmap of planned extensions beyond the current prototype.	48

General Introduction

The intersection of biomedical engineering and industrial automation has produced some of the most impactful engineering achievements of the past two decades. Magnetic resonance imaging, robotic surgery, and closed-loop deep brain stimulation are all examples of automation principles applied in a medical context. Transcranial Magnetic Stimulation (TMS) is another such application, and one where the automation engineering toolbox has not yet been fully exploited.

TMS uses electromagnetic induction to stimulate neurons in the cerebral cortex without surgery or anaesthesia. A time-varying magnetic field, produced by a coil placed on the scalp, induces an electric field inside the brain. When the induced field exceeds the neuronal activation threshold at a given location, the targeted neurons depolarise. By adjusting the coil's position, orientation, and stimulation parameters, clinicians can modulate the activity of specific cortical circuits.

The technology is established. TMS received approval from the US Food and Drug Administration (FDA) in 2008 for treatment-resistant major depression, and has since accumulated clinical indications for OCD, smoking cessation, and post-stroke motor rehabilitation [15]. What remains as an open engineering problem is the real-time prediction of where the stimulation field goes for any given coil placement a computation that currently takes far too long.

This project is carried out in the Department of Automation and Industrial Computing at Aboubakr Belkaïd University, Tlemcen, in collaboration with the neurology department of Tlemcen University Hospital. The collaboration provided clinical context: we observed TMS sessions on real patients, which made the real-time requirement concrete and not abstract.

The project sits at the junction of three active research areas in Automation and Industrial Computing:

- **System identification and surrogate modelling:** replacing expensive physical simulators with fast learned models (Digital Twins).
- **Deep learning for volumetric prediction:** 3D convolutional architectures for spatial field estimation.
- **Real-time industrial supervision:** HMI design principles applied to a medical actuator.

0.1 Problem Statement and motivation

The electric field induced in the brain by TMS depends on the coil position, the coil orientation, and the electrical conductivity structure of the head. Computing this field from first principles requires solving Maxwell’s equations over a finite element mesh of the head. This is the Finite Element Method (FEM), implemented in SimNIBS 4.5.

On a standard Intel Core i5 workstation (CPU only), one FEM simulation takes approximately 180 seconds (3 minutes). In a clinical session, a physician may need to compare a dozen candidate coil positions before targeting the right cortical area. That is 30 minutes of computation for a simple comparison task. Any patient movement during the session requires recalculation. In practice, FEM is used only offline, before the session, not during it.

The motivation is both clinical and scientific. Clinically, more accurate coil positioning means better therapeutic outcomes. A positioning error of 5 mm can change the cortical field intensity by 10–30% depending on the target region [?]. A tool that provides real-time field feedback enables the clinician to correct positioning errors immediately, rather than discovering them after the session.

Scientifically, this problem is a textbook case for surrogate modelling and system identification. The physical system (coil-to-field mapping) is well-defined, expensive to simulate, and smooth enough for neural network approximation. All the tools of Automation – data collection, model training, real-time deployment – transfer directly.

0.2 Research Objectives and methodology

Design, implement, and validate a real-time TMS electric field prediction system based on surrogate modelling and deep learning, using the Digital Twin paradigm from industrial automation.

The project follows the standard Automation and Industrial Computing workflow for system identification and surrogate model deployment:

1. **Problem formalisation:** model TMS as a deterministic input-output system in the sense of control engineering (Chapter 1).
2. **Data collection:** generate $N = 50$ FEM reference simulations automatically using the Fibonacci sphere algorithm for position sampling, Gram-Schmidt orthogonalisation for orientation, and SimNIBS 4.5 for computation (Chapter 2).
3. **Preprocessing:** convert irregular tetrahedral mesh outputs to normalised 64^3 regular voxel grids (Chapter 2).
4. **Model identification:** design and train TMS_CNN_3D, an MLP encoder + 3D ConvTranspose decoder, on 40 simulations using Adam optimisation, MSE loss, and data augmentation (Chapter 3).
5. **Validation:** evaluate on 10 held-out simulations with Pearson R and MAE, and deployment :integrate the trained model into a real-time Matplotlib HMI with parametric sliders and orthogonal field cross-sections, measure full-cycle latency(Chapter 3).
6. **chapter4** integrate the conclusion and the future works).

0.3 Thesis Organisation

General introduction: introduces the context, problem, motivation, and objectives.

Chapter 1 reviews the fundamental concepts (TMS physics, FEM, surrogate modelling, deep learning) and the state of the art in deep learning for TMS field prediction.

Chapter 2 presents the system design: requirements analysis, proposed architecture, data generation pipeline design, and CNN architecture design decisions with UML-style modelling.

Chapter 3 covers the implementation and results: development environment, automated data generation, preprocessing pipeline, CNN training, the real-time supervision interface and discusses the experimental results of this work, its innovations, limitations, and future research directions.

General Conclusion synthesises the achievements, validates the objectives, and charts the path forward.

Chapter 1

Control Engineering Framework for Transcranial Magnetic Stimulation (TMS)

Introduction

Real time supervision of physical processes is a core problem in industrial computing and control engineering. The state of the system is rarely directly observable; it must be inferred from inputs and a model. This project works on exactly that problem, applied to a biomedical actuator: Transcranial Magnetic Stimulation (TMS), where the quantity of interest is the distribution of an electromagnetic field inside brain tissue induced by a coil on the scalp. The engineering question is how to predict that distribution fast enough to support interactive clinical supervision.

The computational bottleneck is severe. The reference calculation method - Finite Element Method (FEM) simulation via SimNIBS - requires approximately three minutes per coil configuration on a standard CPU. This latency is completely incompatible with any real time control or supervision scheme. From a systems engineering standpoint, this is equivalent to a process sensor with a three minute update rate: it cannot support closed loop operation, it cannot support iterative tuning of actuator parameters, and it breaks any feedback architecture that depends on timely process observation.

The solution proposed in this work is surrogate modelling - a well established technique in system identification and Industrial Computer Engineering. A 3D Convolutional Neural Network (CNN) is trained on 50 FEM simulations to learn the input output mapping between coil configuration and induced field volume. Once trained, it reproduces that mapping in approximately 150ms, yielding a $\times 1,200$ speedup and making real time supervision feasible on standard hardware without GPU acceleration. The clinical context - treatment resistant depression, obsessive compulsive disorder, poststroke rehabilitation - shows why the application matters. TMS received FDA approval in 2008 and has since demonstrated significant antidepressant efficacy, with a meta analysis of 2,982 patients showing Hedges' $g = 0.79$ for high frequency stimulation of the left dorsolateral prefrontal cortex [2]. But the therapeutic effectiveness depends directly on targeting precision, and targeting precision depends on knowing where the induced field is - in real time. The clinical requirement therefore reduces to an engineering requirement: sub 200ms field prediction from coil position.

This chapter lays the technical foundations of that engineering requirement. It models

TMS as an input output system in the control engineering sense, describes the electromagnetic physics that govern the coil to field mapping, quantifies the FEM computational bottleneck, and positions the surrogate modelling approach within the current state of the art.

1.1 Physical Characterisation of the TMS Actuator

1.1.1 Electromagnetic Induction Principles and Neuron Activation Mechanism

From a control engineering standpoint, the TMS coil is an actuator that converts an electrical control signal into a spatially distributed physical effect inside a biological medium. Understanding the actuator physics is a prerequisite for any modelling or identification work, so a brief account follows before the system identification formalism is established in Section 1.2.

TMS operates on Faraday’s law of electromagnetic induction. A brief, high intensity current pulse is discharged through a coil placed on the scalp surface, generating a rapidly changing magnetic field of approximately 1-2T. This field crosses the skull without significant attenuation and induces an electric field E (in V/m) in the underlying cortical tissue - the actuator’s output. Introduced to clinical practice by Barker et al. in 1985 [1], TMS requires no physical contact and no anaesthesia, which simplifies the integration of positioning hardware. The induced field acts directly on cortical neuron membranes by modifying their transmembrane potential. When field intensity at a given location exceeds the motor threshold, depolarisation occurs - neurons activate or are inhibited depending on stimulation parameters. In control terms, the spatial distribution of E determines which cortical region is engaged and at what intensity: it is the process output of interest. This distribution is governed by Maxwell’s equations in quasi static approximation and is shaped by two actors - coil geometry and orientation, and the head’s layered electrical conductivity structure. Conductivity discontinuities between scalp, skull, cerebrospinal fluid, grey matter, and white matter create concentration effects at tissue interfaces, a detail that complicates numerical modelling and surfaces again in the discussion of prediction errors in Chapter 3.

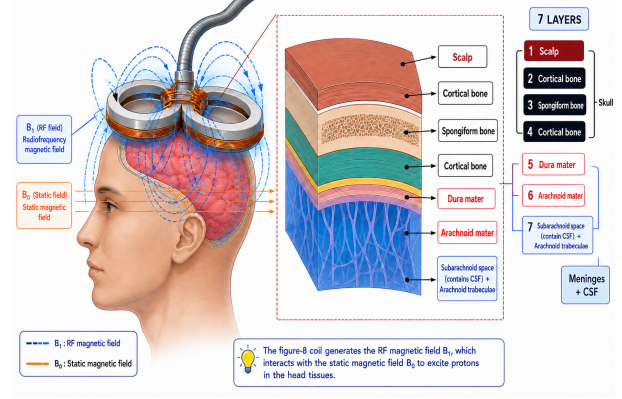


Figure 1.1: Input and output data used for training.

1.1.2 Coil Geometry and Spatial Selectivity of the Actuator Output

The spatial selectivity of the TMS actuator - how narrowly the induced electric field can be focused on a target cortical region - is determined primarily by the coil geometry. This actuator design parameter has direct implications for the modeling input space.

Among the available coil configurations (circular, H coil, and double cone), the figure-8 coil has become the standard in both clinical practice and research due to its superior spatial selectivity. It consists of two adjacent circular windings carrying currents in opposite directions. At their central junction, the magnetic fields generated by each wing combine constructively, concentrating the stimulation area to approximately 2cm^2 beneath the coil center. This high focality enables targeted cortical stimulation and makes feedback based positioning strategies technically feasible.

The coil used in this project is the Magstim 70mm figure-8 model. Its physical definition file (Magstim_70mm_Fig8.ccd), provided by the manufacturer, is directly loaded into SimNIBS for electric field computation. The coil contains nine turns per wing, with an outer radius of 44mm and an inner radius of 26 mm.

From a systems identification perspective, effective stimulation depends strongly on the coil orientation relative to the targeted cortical structures. A positioning error of approximately 7.8° in roll or pitch, commonly encountered during manual placement, can significantly reduce the induced cortical electric field [10]. In addition, a targeting error of 5mm may alter the field intensity by 10 – 30% [12]. These observations demonstrate the high sensitivity of TMS stimulation to both coil position and orientation.

Furthermore, slow field prediction prevents real time adjustment of stimulation parameters. Consequently, electric field estimation must be performed in less than 200ms for each coil position to support practical closed loop operation. This sub-200 ms requirement constitutes a key design objective of the present project.

Because of this sensitivity and real time constraint, the complete coil configuration - including both position and orientation - must be encoded in the CNN input vector. Representing position alone would provide an incomplete description of the stimulation

setup and would limit the model's ability to accurately predict the induced cortical electric field.

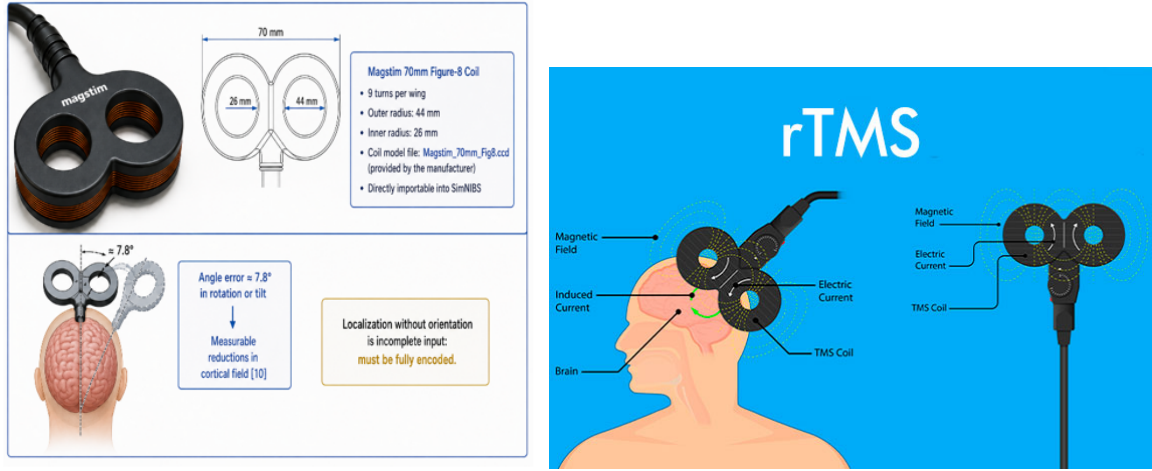


Figure 1.2: (a) The TMs used in this project, (b) The principle of TMS coil

1.2 Input Output Modelling of the TMS System

The central contribution of this section is to formalise TMS within the deterministic input output framework of control engineering. This is not a cosmetic reframing - it justifies the use of system identification methods, defines what the CNN must learn, and specifies what "real time" means in this context.

1.2.1 System Input: Coil Configuration as a State Vector

The system input is the coil's complete spatial configuration: position (x, y, z) in millimetres on the skull surface plus full three dimensional orientation. In control terms this is the setpoint the operator adjusts to steer the actuator output toward a cortical target. Section 1.1.2 already showed why orientation matters as much as position.

$$\mathbf{M} = [\mathbf{x}_{\text{dir}} \ \mathbf{y}_{\text{dir}} \ \mathbf{n} \ \mathbf{p}] \in \mathbb{R}^{4 \times 4} \quad \mathbf{T} = \begin{bmatrix} x_x & y_x & n_x & p_x \\ x_y & y_y & n_y & p_y \\ x_z & y_z & n_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\mathbf{x}_{\text{dir}}$ and $\mathbf{y}_{\text{dir}}$ are orthogonal tangential vectors on the skull surface, $\mathbf{n}$ is the unit normal pointing inward, and $\mathbf{p}$ is the coil centre position in millimetres. Flattened to 16 scalar values, this matrix is the CNN's input. The encoding is lossless: all the geometric information needed to determine the induced field is preserved.

Table 1.1: Composition of the 4×4 homogeneous transformation matrix (CNN input).

Column	Vector	SimNIBS role	Description
0	$\mathbf{x}_{\text{dir}}$ (tangent 1)	Tangential direction 1	1st Gram Schmidt tangent
1	$\mathbf{y}_{\text{dir}}$ (tangent 2)	pos_ydir (required)	SimNIBS orientation parameter
2	$\mathbf{n}$ (unit normal)	Axis toward brain	Normal to skull surface
3	$\mathbf{p}$ (position)	Coil centre	Coordinates in mm on the scalp

1.2.2 System Output: Volumetric Electric Field Distribution

The system output is the three dimensional electric field $\mathbf{E}(\mathbf{r})$ induced throughout the brain tissue. In our implementation, this is represented as a discrete volume of $64 \times 64 \times 64$ voxels (262,144 values total), covering a region of interest of 150mm centred on the spherical brain model. Each voxel stores the field magnitude $\|\mathbf{E}\|$ in V/m, normalised to $[0, 1]$ for training. This volumetric format is functionally analogous to a 3D greyscale image: it encodes stimulation intensity at every point in the tissue. The non-zero voxels (approximately 49% of the total) correspond to the brain region; zero valued voxels represent air or scalp exterior to the brain.

1.2.3 The Physical Transfer Operator: Forward and Inverse Problems

The relationship between coil configuration and field distribution can be written compactly as:

$$\mathbf{E}(\mathbf{r}) = \mathbf{F}(\mathbf{M}_{\text{coil}}, \mathbf{G}_{\text{head}}) \quad (1.2)$$

where $\mathbf{G}_{\text{head}}$ encodes the geometry and electrical properties of the head, and $\mathbf{F}$ is the physical operator implicitly defined by solving Maxwell’s equations on the head mesh through FEM. This gives rise to two complementary subproblems. The forward problem - computing $\mathbf{E}$ for a given $\mathbf{M}_{\text{coil}}$ - is precisely what our 3D CNN learns to approximate. The inverse problem - finding the coil configuration that maximises stimulation at a target cortical location while minimising spread - is considerably harder and lies beyond the scope of the present work, though it represents a direct and natural extension.

Table 1.2: TMS modelled as an input output system within the control engineering frame

Component	Description	Value in this project
System input	Coil position and orientation	4×4 matrix - 16 scalars
Physical operator $\mathbf{F}$	Maxwell’s equations (FEM)	SimNIBS 4.5 - tetrahedral solver
System output	$\mathbf{E}(\mathbf{r})$ in V/m	64^3 voxel volume, $[0, 1]$
Head model	Scalp geometry and conductivities	$R = 85\text{mm}$, brain $R = 70\text{mm}$
Coil	Winding geometry	Magstim 70mm figure-8 (.ccd)
Surrogate model	Fast approximation of $\mathbf{F}$	3D CNN - inference $\approx 150\text{ms}$

1.3 Computational Constraints: The FEM Bottleneck

Clear as the input output description is, evaluating $\mathbf{F}$ through the reference approach - FEM - remains computationally prohibitive. On a standard workstation (Intel Core i5 7th generation, CPU only), each SimNIBS simulation requires approximately three minutes. This is incompatible with any interactive clinical workflow.

Interactive use requires evaluating many coil positions: if the patient moves, if several candidate positions need comparing, or if fine angular adjustments are being explored, a three minute wait per evaluation is simply untenable. The cost is structural. A tetrahedral head mesh has 113,912 elements in our spherical model, each requiring the coupled solution of the conductivity equations. There is no shortcut that preserves physical accuracy without paying that cost.

Table 1.3: Comparison of computation times across clinical scenarios: FEM versus the 3D CNN surrogate.

Clinical scenario	FEM time	CNN time	Feasible?
1 coil position	$\approx 180,000\text{ms}$	$\approx 150\text{ms}$	Yes (CNN only)
10 candidate positions	$\approx 30\text{min}$	$\approx 1.5\text{s}$	Yes (CNN only)
Correction after head movement	$\approx 3\text{min each}$	$< 200\text{ms}$	Yes (CNN only)

The speed ratio is 1,200:1 in favour of the neural network. That bottleneck is the whole motivation for this project.

1.4 State of the Art: Surrogate Models and Deep Learning in Neurostimulation

1.4.1 Surrogate Modelling in Control Engineering

System identification aims to build a mathematical model of a process from observed input output data. When the physical system is computationally expensive to evaluate repeatedly, a common approach is to develop a surrogate model, also known as a meta-model or emulator, which approximates the input output behaviour of the original system at a much lower computational cost.

In our setting, the physical system is the SimNIBS finite element method (FEM) solver. The proposed 3D CNN acts as a surrogate model: trained on 50 FEM simulations, it learns the mapping between coil configurations and the resulting electric field distribution, reproducing FEM predictions in approximately 150 ms instead of 180,000 ms. This paradigm is closely related to the concepts of Industry 4.0 and the Digital Twin, where a lightweight digital representation emulates the behaviour of a physical system in near real time.

1.4.2 Deep Learning Approaches to E Field Prediction

Yokota et al. (2019) [4] were among the first to apply deep learning to TMS E field estimation. Their 3D U-Net, trained on 261,072 simulation pairs across 37 subjects,

achieved correlation coefficients above $R = 0.93$ with a GPU inference time of 24ms. Our approach differs fundamentally: no real MRI is available, the head model is spherical, and the CNN input is the 16 scalar position-orientation matrix rather than full volumetric anatomy.

Sathi et al. (2021) [3] published a deep neural network trained on 100 COMSOL simulations for predicting the E field from six coil design parameters. Their test set results of $R^2 = 0.766$ and a 0.04 s inference time (vs. 6min 15 s for COMSOL) provided early evidence for the viability of a surrogate DNN approach at a speed gain comparable to what we report here.

Berger et al. (2025) [5] published the first systematic evaluation of a modified 3D U-Net for whole brain E field estimation across 100 HCP MRI subjects with 15,678 coil configurations each. Their results - mean DICE of 0.71 ± 0.06 , spatial correlation of 0.79, and an inference time of 0.8 s - confirm that accuracy and speed can be achieved at scale, though at the cost of substantially larger datasets and GPU infrastructure.

Moser et al. (2024) described the first network that simultaneously predicts optimal TMS coil placement and the associated field distribution. Their multitask Attention U-Net, trained on 18,270 SimNIBS optimisations across 105 subjects, reached a position MAE below 1.8mm, E field correlation above 97%, and an inference time of 35ms on GPU. This is the direction toward which our work could naturally evolve - our CNN currently solves only the forward problem, leaving all positioning decisions with the clinician.

Table 1.4: Comparative summary of deep learning approaches to TMS E field prediction.

Reference	Architecture	Dataset	Metric	Inf. time	Head model
Yokota 2019	3D U-Net	37 subj. 261k simul.	$R = 0.93 - 0.97$	0.024 s (GPU)	Realistic (MRI)
Sathi 2021	DNN (512-64)	100 simul.	$R^2 = 0.766$	0.04 s (GPU)	Spherical
Li 2022	Self-supervised U-Net	125 subjects	Corr. = 0.978	1.47 s (CPU)	Realistic + spherical
Berger 2025	Modified 3D U-Net	100 subj. 15k/subj.	DICE = 0.71	0.8 s	Realistic (MRI)
Moser 2024	Multi-task Att. U-Net	105 subj. 18k simul.	Corr. > 97%	35 ms (GPU)	Realistic (MRI)

1.4.3 Connection to Control and Industrial Computing

The problem treated here fits squarely within system identification as the field defines it. The system is the coil to field mapping: a nonlinear function from a finite dimensional input (16 scalars) to a high dimensional volumetric output (262,144 values). Training the 3D CNN is formally equivalent to generalised nonlinear least squares identification, minimising MSE over a set of input output observations.

The supervision interface developed in Chapter 4 constitutes a real time control system in the industrial sense: the operator adjusts setpoint parameters (coil position and orientation), the system computes and displays the process response (field E) in near real time, enabling informed open loop decision making. This closely resembles the Human Machine Interface architecture found in industrial SCADA systems. Looking further ahead, a fully differentiable position-to-field mapping opens the way to inverse optimisation by gradient descent - the first step toward an automated TMS positioning system.

To better understand this integration, the general architecture of the proposed semi-closed control loop is mapped out in Figure 1.3. This layout illustrates how the 3D CNN operates as a live surrogate model, bypassing the heavy processing delays inherent in traditional FEM techniques to provide the clinician with instant visual feedback. The dynamic operation of the proposed semi-closed control framework functions through a

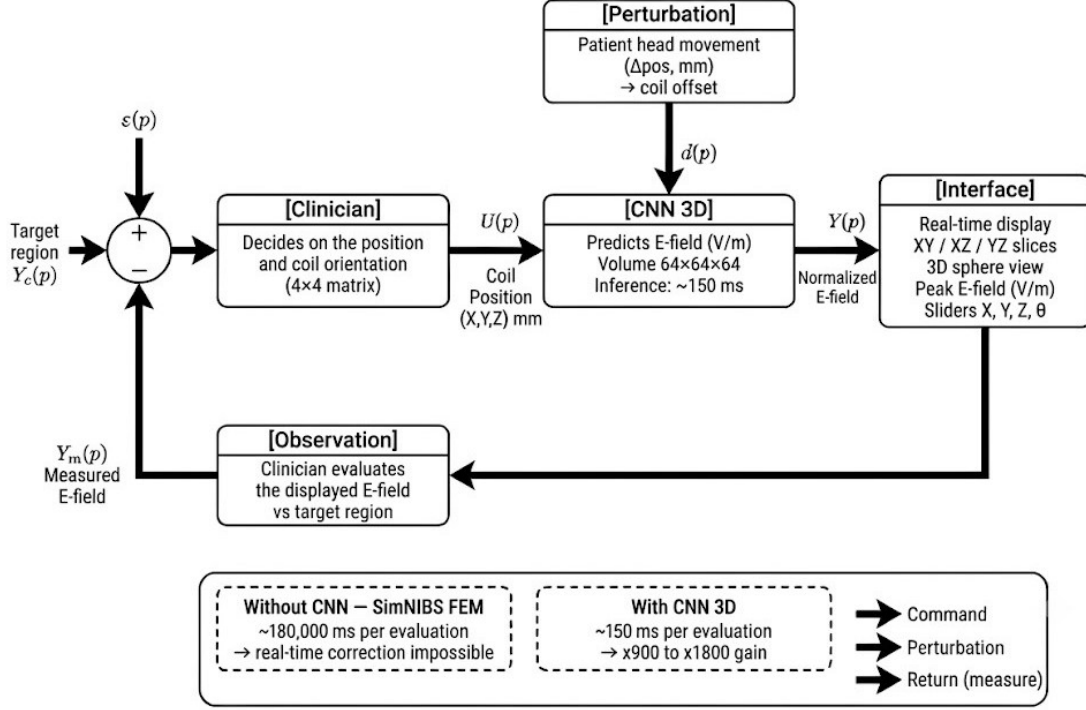


Figure 1.3: Block Diagram - TMS system assisted by CNN

sequential, closed-loop stabilization process described as follows:

First, the procedure begins at the comparator, where the clinician defines the target cortical region by setting the desired volumetric electric field distribution required for the therapeutic session. Any discrepancy between this target profile and the actual feedback yields an error signal. Operating as the core feedback controller, the clinician perceives this system deviation and manipulates the command vector, actively repositioning and adjusting the spatial coordinates (X, Y, Z) and the angular orientation (theta) of the physical TMS coil.

During this manual manipulation, external perturbations primarily driven by sudden or gradual patient head movements interfere with the physical setup, directly modifying the actual spatial relationship between the actuator and the cortex. To counteract this, the altered geometric state is fed into the 3D CNN, which acts as the system plant model. Bypassing heavy numerical solvers, this surrogate estimator evaluates the localized physical interactions to predict the entire 64x64x64 induced volumetric field distribution within a real-time window of approximately 150 ms.

Finally, the predicted field map is immediately processed by the graphical unit interface, displaying live structural slices to the operator. By inspecting this visual output through direct observation, the clinician dynamically evaluates the spatial alignment, routing the observed field state back as a measured feedback signal to recompute the tracking error, thereby manually closing the control loop to ensure precise target tracking.

1.5 Gap Analysis and Project Objectives

1.5.1 Identified Gaps in the Literature

Analysis of existing work reveals three gaps that this project is designed to address. First, virtually all deep learning approaches to E field prediction depend on large simulation datasets and GPU infrastructure. No prior work, to our knowledge, has validated this class of approach under genuinely constrained conditions: CPU only, 50 simulations. Second, existing work concentrates on prediction performance metrics without integrating an operator facing real time interface for interactive exploration of the coil position space. Third, a complete, documented pipeline from automated FEM dataset generation through to a working real time interface is absent from the academic literature in the context of an industrial automation and computing dissertation.

1.5.2 Primary Objective and Contributions

The main objective is to design and validate a 3D CNN surrogate model capable of predicting the TMS induced electric field in under 200ms from the coil position and orientation, replacing an FEM computation that takes approximately 3 minutes. The CNN addresses only the forward problem; the clinical positioning decision remains entirely with the practitioner.

Table 1.5: Summary of the three project contributions and the results obtained.

Contribution	Description	Result
C1 Dataset	Automated pipeline of 50 SimNIBS simulations	3.28GB, 50 normalised 64^3 volumes
C2 3D CNN	MLP encoder + ConvTranspose3D decoder	$R = 0.887$, $MAE = 0.022$, ≈ 150 ms
C3 Interface	Real time supervision: sliders + E field cross sections	Inference < 200 ms, working prototype

1.6 Conclusion

This chapter has shown that TMS can be modelled as an input output system in which a 4×4 configuration matrix maps to a 64^3 electric field volume. It has also shown that the reference FEM solver, at approximately 180,000ms per simulation, is fundamentally incompatible with interactive clinical use. Existing deep learning surrogate models show that acceleration factors of $\times 1,000$ to $\times 97,000$ are achievable - but only under resource conditions well beyond what most clinical settings can provide. Our contribution targets precisely the opposite end of that spectrum: limited hardware, limited data, and a focus on producing a usable prototype rather than a benchmark beating system. The next chapter describes how the training dataset was constructed.

Chapter 2

System Identification and Data Generation

Introduction

Building a surrogate model is only as good as the data it is trained on. This chapter describes the complete methodology used to generate and prepare the training dataset for the CNN, following the system identification principles studied in Automatic Control Engineering - where input output data are first collected from the physical system before any modelling takes place. Here, the physical system is the SimNIBS FEM solver, and the data collection process required approximately two and a half hours of continuous CPU computation.

The pipeline proceeds through four main stages: surrogate modelling strategy, reference FEM modelling, dataset generation protocol, and preprocessing with normalisation.

2.1 Surrogate Modelling Strategy

A Digital Twin is a digital representation of a physical system that is sufficiently accurate to reproduce its behaviour and operate in place of the real system during supervision, optimisation, fault diagnosis, or predictive control. The concept is widely used in industrial automation, where evaluating the real system is computationally expensive and a faster model is required to answer repeated queries in real time. In this project, the trained 3D CNN serves as the Digital Twin of the TMS system. It represents the complete physical chain (coil + head + FEM solver) and predicts the induced electric field for any coil position in approximately 150 ms.

The construction of this Digital Twin follows the standard three step system identification procedure: observation generation, model training, and validation.

First, a dataset of ($N = 50$) FEM simulations is generated using SimNIBS, producing 50 input output pairs $((\mathbf{M}_{\text{coil},i}, \mathbf{E}_i))$, where $\mathbf{M}_{\text{coil},i}$ denotes the coil configuration and $\mathbf{E}_i$ the corresponding electric field distribution.

Second, the 3D CNN is trained on 80% of these observations (40 simulations) by minimising the Mean Squared Error (MSE) with respect to the FEM ground truth.

Finally, the remaining 20% of the data (10 simulations), withheld throughout training, are used as a validation set to assess the model's generalisation capability.

From a system identification perspective, this approach corresponds to a nonpara-

metric neural network based model, equivalent to a static instance of the NLARX (Non-Linear AutoRegressive with eXogenous input) framework, applied here without temporal dynamics.

2.2 Reference Modelling by Finite Elements

2.2.1 Spatial Discretisation and Tissue Conductivities

Simulating the TMS field means solving Maxwell’s equations in quasi static approximation, which reduces to the generalised Laplace equation:

$$\nabla \cdot (\sigma \nabla \phi) = 0, \quad \mathbf{E} = -\nabla \phi, \quad \mathbf{J} = \sigma \mathbf{E} \quad (2.1)$$

where ϕ is the electric potential, σ is the tissue conductivity, and $\mathbf{J}$ is the current density. The FEM solves this numerically by dividing the head volume into tetrahedral elements, each assigned a conductivity value corresponding to its tissue type.

Table 2.1: Electrical conductivities of head tissues in the full SimNIBS model. In our simplified spherical model, only scalp and grey matter are used.

Tissue	Conductivity (S/m)	SimNIBS label
Scalp	0.465	Label 5
Skull bone	0.010	Label 4
Cerebrospinal fluid (CSF)	1.654	Label 6
Grey matter	0.275	Label 2
White matter	0.126	Label 1

In our simplified model, only scalp ($\sigma = 0.465\text{S/m}$) and brain grey matter ($\sigma = 0.275\text{S/m}$) are included. This simplification is deliberate: validating the CNN approach on a controlled geometry is the necessary first step before tackling the complexity of a realistic multitissue head model.

2.2.2 The SimNIBS 4.5 Simulation Environment

SimNIBS (Software for Non Invasive Brain Stimulation) is the international reference pipeline for finite element modelling of transcranial stimulation. It covers the full processing workflow: MRI segmentation, tetrahedral mesh generation, coil placement, numerical solution of Maxwell’s equations, and result visualisation.

In this project, SimNIBS 4.5 takes three inputs: the tetrahedral head mesh file (`my_sphere_msh_Z.msh`), the coil physical description file (`Magstim_70mm_Fig8.ccd`), and the 4×4 transformation matrix defining the coil’s position and orientation. It produces an enriched `.msh` file containing the electric field vector $\mathbf{E}$ at every mesh element - the supervised learning ground truth.

The head model is a two layer spherical construction: an outer sphere (scalp, radius 85mm, Label 5) and a concentric inner sphere (brain, radius 70mm, Label 2), shifted upward by 10.625mm along the Z axis to approximate the anatomical position of the brain within the skull. This model is generated from a $101 \times 101 \times 101$ NIfTI voxel

grid at 2.125mm voxel resolution and converted to a tetrahedral mesh using SimNIBS's meshmesh command.

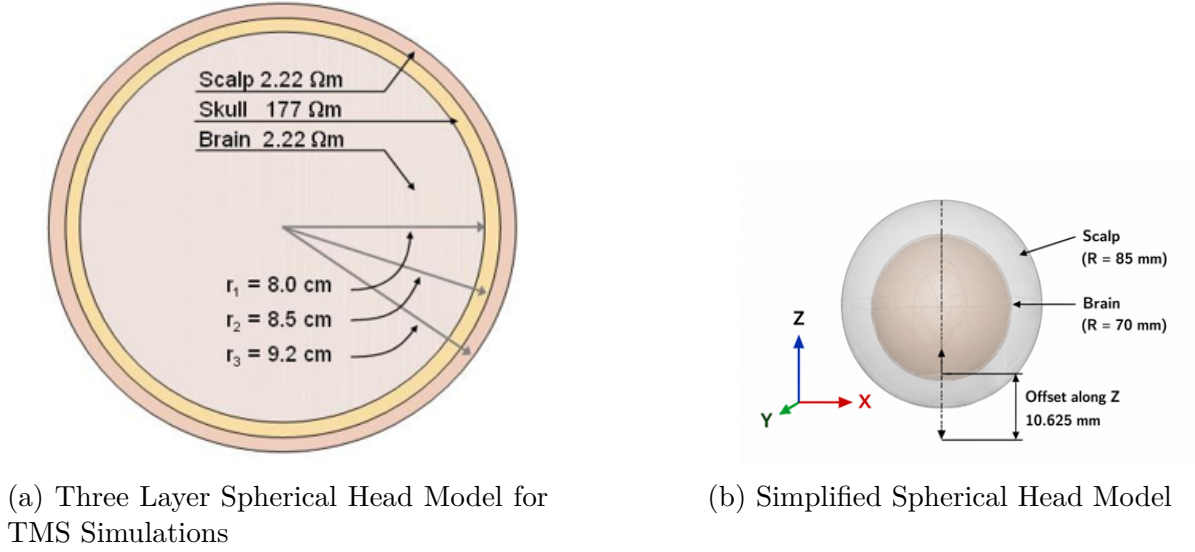


Figure 2.1: Head Modeling Approaches for TMS Simulations

2.3 Dataset Generation Protocol

Full automation of the data generation pipeline was a non negotiable design requirement: manual management of 50 simulation runs would have introduced inconsistencies and was practically unfeasible. The Python script `phase1_z.py` handles the entire process - position definition, orientation computation, SimNIBS script generation, simulation execution, logging, and result verification.

The 50 coil positions are distributed across the upper hemisphere ($Z \geq 0$) using the Fibonacci sphere algorithm [16], which provides near uniform angular coverage without the pole clustering that afflicts regular latitude longitude grids [9]. For each position on the spherical surface, the coil normal is aligned with the skull surface normal using Gram Schmidt orthogonalisation, ensuring a consistent and reproducible orientation convention across all simulations.

Table 2.2: Dataset composition after generation (raw SimNIBS outputs)

Files	Count	Size
<code>positions_Z.csv</code>	1 file	< 1MB
<code>sim_001.msh</code> <code>sim_050.msh</code>	50 files	1265MB each
<code>sim_001.log</code> <code>sim_050.log</code>	50 files	< 2KB each
Total	350 files	3.28GB

Average simulation time was 180 s per coil position on the project hardware, giving a total of approximately 2 h 30 for the full dataset. This figure makes the computational bottleneck of FEM tangible - and makes the case for a surrogate model concrete.

Algorithm 1 TMS Simulation Dataset Generation

- 1: **Input:** Parameters for head model and simulation settings
 - 2: Generate coil positions using Fibonacci sphere sampling (50 positions, $Z \geq 0$)
 - 3: Define the spatial position of each coil
 - 4: Compute coil orientations using Gram–Schmidt orthogonalization
 - 5: Automatically generate SimNIBS scripts for each configuration
 - 6: Execute simulations (50 FEM runs)
 - 7: Perform logging and monitoring of each simulation
 - 8: Verify and validate simulation results
 - 9: **Output:** Validated dataset
-

2.4 Signal Preprocessing and Normalisation

The .msh files produced by SimNIBS define data on an irregular tetrahedral mesh of 113,912 elements and 33,150 nodes. The CNN requires a regular 3D voxel grid. Bridging this format incompatibility was the main preprocessing challenge, addressed by a seven-step pipeline implemented in `phase2_preprocessing.py`.

Table 2.3: Preprocessing pipeline: irregular mesh 64^3 voxel grid

Step	Operation	Tool
1 Reading	Load SimNIBS .msh file	<code>simnibs.read_msh()</code>
2 Extraction	Isolate brain region (Label 2)	<code>crop_mesh(tags=2)</code>
3 Field	Extract $\ \mathbf{E}\ $ magnitude	<code>brain.field['normE']</code>
4 Resampling	Elements nodes (weighted average)	Custom Python
5 Interpolation	Resample onto 64^3 regular grid	<code>scipy.interpolate.griddata</code>
6 Normalisation	Clip at 99.9th percentile, divide	NumPy
7 Saving	Store as float32 NumPy array	<code>np.save()</code>

One technical subtlety deserves mention. SimNIBS computes the electric field on tetrahedra (113,912 values) rather than on mesh nodes (33,150 values). Converting from element centred to node centred values by weighted average is therefore necessary before the interpolation onto the regular grid can proceed correctly.

The final normalisation to $[0, 1]$ - using the 99.9th percentile of each volume as the local maximum - is consistent with the Sigmoid activation at the CNN output, which produces values in the same range. The global maximum field value observed across the dataset (4.5 V/m) is preserved in metadata for back conversion to physical units during inference in the following sections.

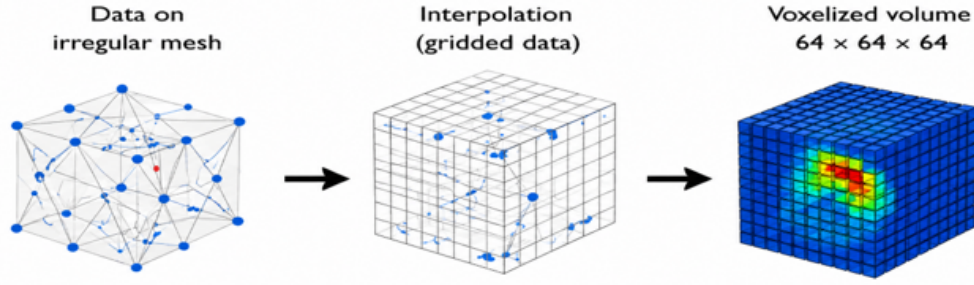


Figure 2.2: Preprocessing pipeline illustration: (a) irregular tetrahedral mesh output from SimNIBS, (b) brain region extraction, (c) interpolated 64^3 voxel grid, (d) normalised field volume.

Table 2.4: Parameters of the interpolation ROI grid.

Parameter	Value	Justification
Output resolution	$64 \times 64 \times 64$ voxels	Standard for 3D CNN E field work
ROI centre	$[0, 0, 10.625]$ mm	Brain geometric centre (Z offset)
ROI size	150 mm	Covers scalp sphere ($R = 85\text{mm}$) with margin
File size (.npy)	$\approx 1\text{MB}$ per simulation	Float32, 64^3
Non zero voxels	$\approx 49\%$ ($< 129,630/262,144$)	Brain region only
Maximum field	4.5 V/m	Used for back conversion in Chapter 4

2.5 Dataset Overview

A dataset of 50 simulations was generated to train and validate the CNN model. The input data consist of flattened 4×4 matrices, while the outputs are normalized 64^3 volumetric distributions. Preprocessing reduced the dataset size from 3.28 GB to approximately 50 MB, improving storage efficiency. The data were split into 80% for training and 20% for validation. Simulations were performed using SimNIBS 4.5 and a Magstim 70 mm Figure-8 coil, with stimulation positions uniformly distributed over the upper hemisphere using the Fibonacci sphere method [16]. Although the dataset provides adequate spatial coverage, increasing the number of simulations and using more realistic head models could further improve model performance.

Table 2.5: Complete identification dataset summary

Parameter	Value
Number of simulations	50
CNN input format	Flattened 4×4 matrix + vector of 16 values
CNN output format	Float32 volume $64 \times 64 \times 64$ normalised $[0, 1]$
Total Phase 1 time	$\sim 2\text{h}30$ ($50 \times 180\text{s}$ on i5 CPU)
Raw dataset size	3.28 GB (outputs/ folder)
Preprocessed dataset size (.npz)	~ 50 MB ($50 \times \sim 1$ MB)
Training / validation split	40 simulations (80%) / 10 simulations (20%)
Spatial distribution	Fibonacci sphere + upper hemisphere $Z \geq 0$
Head model	Two-layer spherical (scalp $R = 85$ mm, brain $R = 70$ mm)
FEM simulation tool	SimNIBS 4.5 — finite element solver
Coil model	Magstim_70mm_Fig8.ccd — figure-8 70 mm coil

Table 2.6: Parameters of the interpolation ROI grid.

Parameter	Value	Justification
Output resolution	$64 \times 64 \times 64$ voxels	Standard for 3D CNN E field work
ROI centre	$[0, 0, 10.625]$ mm	Brain geometric centre (Z offset)
ROI size	150 mm	Covers scalp sphere ($R = 85\text{mm}$) with margin
File size (.npz)	$\approx 1\text{MB}$ per simulation	Float32, 64^3
Non zero voxels	$\approx 49\%$ ($< 129,630/262,144$)	Brain region only
Maximum field	4.5 V/m	Used for back conversion in Chapter 4

The dataset of 50 simulations is deliberately smaller than those used in the reference literature - it is a methodological validation on a spherical prototype under constrained resources, sufficient to show that the surrogate approach works. Moving to 150-300 simulations on a realistic MRI derived model is the natural next step for a clinically deployable version.

2.6 Architecture of the 3D Neural Network

The CNN architecture must solve a fundamental challenge: transform a compact input vector of 16 scalars (the flattened 4×4 homogeneous transformation matrix encoding coil position and orientation) into a large output volume of $64^3 = 262,144$ voxels (the normalised electric field distribution). This dimensionality ratio of 1:16,384 dictates an asymmetric encoder decoder architecture.

The chosen design combines an MLP encoder overcomplete suited to processing a small scalar input with no spatial structure and a 3D convolutional decoder based on

transposed convolutions (ConvTranspose3d), capable of progressively reconstructing a volumetric output.

2.6.1 MLP Encoder overcomplete: Processing the Geometric Input

The encoder overcomplete is a Multi-Layer Perceptron with three successive linear layers. Its role is to project the 16 dimensional input vector into a 256 dimensional latent representation that implicitly captures the relationship between coil position/orientation and the resulting spatial field distribution. Each linear layer is followed by a LeakyReLU ($\alpha = 0.2$) activation, chosen to preserve a non zero gradient for negative pre activations - avoiding the dying ReLU problem that is particularly common with normalised inputs close to zero.

A Dropout regularisation (rate = 0.2) is applied after the first two encoder layers. This mechanism randomly deactivates 20% of neurons at each training step, forcing the network to build redundant internal representations and thereby reducing the risk of overfitting - a critical consideration given the small training set of 40 examples.

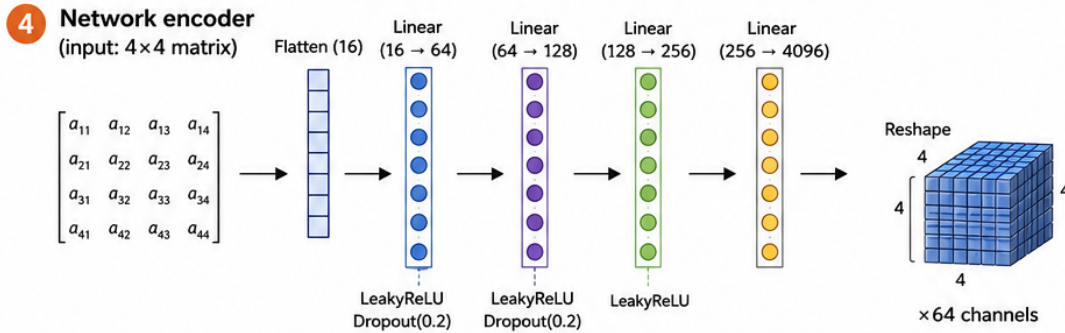


Figure 2.3: network MLP encoder overcomplete

Code extract - MLP encoder (phase3_cnn.py):

```
self.encoder = nn.Sequential(
    nn.Linear(16,64), nn.LeakyReLU(0.2), nn.Dropout(0.2),
    nn.Linear(64,128), nn.LeakyReLU(0.2), nn.Dropout(0.2),
    nn.Linear(128,256), nn.LeakyReLU(0.2),
)
self.fc_to_3d = nn.Linear(256, 4*4*4*64) # 256 -> 4,096 values
```

2.6.2 3D Projection Layer

A linear projection layer (fc_to_3d) transforms the 256 dimensional latent vector into a tensor of 4,096 values ($256 \rightarrow 4 \times 4 \times 4 \times 64$), which is then reshaped into a 3D mini volume of shape (B, 64, 4, 4, 4). This layer forms the bridge between the scalar domain of the encoder and the volumetric domain of the convolutional decoder. The choice of an initial 4^3 resolution is motivated by the parameter budget: this single layer accounts

for 1,052,672 parameters (45.6% of the total network), concentrating representational capacity at the transition point.

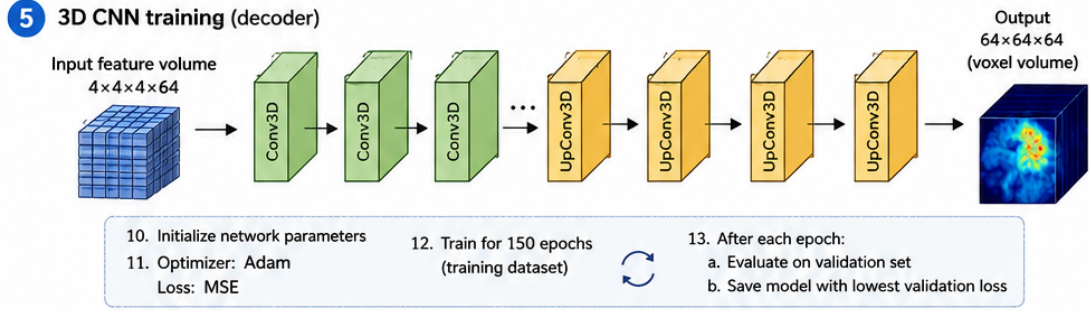


Figure 2.4: The architecture of the 3DCNN decoder

2.6.3 3D Convolutional Decoder: Progressive Upsampling

The decoder consists of four successive ConvTranspose3d blocks, each doubling the spatial resolution of the volume ($4^3 \rightarrow 8^3 \rightarrow 16^3 \rightarrow 32^3 \rightarrow 64^3$), followed by a final Conv3d output layer.

Transposed convolutions perform the reverse operation of a standard convolution: they increase spatial resolution while applying learnable filters. The parameter stride = 2 guarantees a $\times 2$ resolution increase at each stage. The combination kernel_size = 4, padding = 1 is chosen to minimise checkerboard artefacts common in generative networks. Each ConvTranspose3d block is followed by BatchNorm3d (which normalises activations across the batch, stabilising training with small batch sizes $B = 4$) and LeakyReLU(0.2). The output layer Conv3d($16 \rightarrow 1$, kernel = 3, padding = 1) reduces the channel dimension to a single scalar field, followed by a Sigmoid activation that constrains the output to $[0, 1]$ consistent with the normalisation performed in Phase 2.

Table 2.7: Detailed architecture of the 3D CNN TMS_CNN_3D (2,308,673 total parameters)

Block	Layer	Input	Output	Parameters
MLP Encoder OC	Linear(1664) + LReLU + Dropout	(B,16)	(B,64)	1,088
MLP Encoder OC	Linear(64128) + LReLU + Dropout	(B,64)	(B,128)	8,320
MLP Encoder OC	Linear(128256) + LReLU	(B,128)	(B,256)	33,024
3D Projection	Linear(2564096) + reshape	(B,256)	(B,64,4s)	1,052,672
3D Decoder	ConvTranspose3d(64128, k=4,s=2,p=1)+BN+LReLU	(B,64,4s)	(B,128,8s)	524,544
3D Decoder	ConvTranspose3d(12864, k=4,s=2,p=1)+BN+LReLU	(B,128,8s)	(B,64,16s)	262,208
3D Decoder	ConvTranspose3d(6432, k=4,s=2,p=1)+BN+LReLU	(B,64,16s)	(B,32,32s)	65,568
3D Decoder	ConvTranspose3d(3216, k=4,s=2,p=1)+BN+LReLU	(B,32,32s)	(B,16,64s)	16,400
Output	Conv3d(161, k=3,p=1) + Sigmoid	(B,16,64s)	(B,1,64s)	433

Table 2.8: Justification of the architectural choices of the 3D CNN

Architectural choice	Justification	Rejected alternative
MLP encoder OC (3 layers)	Scalar input (16 values) — no spatial structure to exploit	CNN on input (not applicable)
ConvTranspose3d (4 blocks)	Standard approach for progressive 3D volume generation	Bilinear upsampling (artefacts)
LeakyReLU ($\alpha = 0.2$)	Prevents dying ReLU on small, normalised datasets	Classic ReLU
Dropout (0.2)	Regularisation with only 40 training examples	L2 weight decay alone
BatchNorm3d	Stabilises training with small batch size ($B = 4$) on CPU	LayerNorm (unsuitable for 3D)
Sigmoid (output)	Consistent with $[0,1]$ normalisation from Phase 2	ReLU (unbounded output)
2,308,673 parameters	Sufficient capacity for 50 CPU generated simulations	U-Net (excessive memory on CPU)

2.7 Non Linear Identification: Cost Function and CPU Optimisation

From a control engineering perspective, training the 3D CNN constitutes a non linear parametric identification procedure: the objective is to find the parameter set θ (2,308,673 weights and biases) that minimises the mean squared error between CNN predictions and FEM reference outputs across all 40 training pairs.

2.7.1 Cost Function: MSE on 3D Volumes

The cost function is the Mean Squared Error (MSE), computed voxel by voxel across the 262,144 output values:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_i \left\| \hat{E}(\mathbf{r}_i; \theta) - E_{\text{FEM}}(\mathbf{r}_i) \right\|^2$$

where $\hat{E}(\mathbf{r}_i; \theta)$ is the CNN prediction for simulation i , and $E_{\text{FEM}}(\mathbf{r}_i)$ is the FEM ground truth. MSE is chosen for its sensitivity to large localised errors, which encourages faithful reconstruction of field intensity maxima — clinically critical information for identifying the cortical stimulation zone.

2.7.2 Optimiser and Learning Rate Scheduling

Optimisation is performed using the Adam algorithm (Adaptive Moment Estimation), with initial learning rate $\eta = 10^{-3}$ and standard parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$. Adam is preferred over SGD because it automatically adapts the per parameter learning rate from gradient history — a key advantage with small batch sizes ($B = 4$).

A ReduceLROnPlateau scheduler halves the learning rate when the validation loss shows no improvement for 15 consecutive epochs (patience = 15, factor = 0.5), enabling progressive refinement without oscillations near convergence.

Code extract training configuration (phase3_cnn.py):

```

criterion = nn.MSELoss()
optimizer = optim.Adam(
    model.parameters(), lr=1e-3, weight_decay=1e-4)
scheduler = optim.lr_scheduler.ReduceLROnPlateau(
    optimizer, mode="min", factor=0.5, patience=15 )

```

2.7.3 Training Hyperparameters

This table presents the main training hyperparameters used for the 3D CNN model. It also explains the choice of each parameter in the training setup.

Table 2.9: Training hyperparameters of the TMS 3D CNN

Hyperparameter	Value	Justification
Input size (INPUT_DIM)	16	4×4 matrix flattened to vector
Output size	$64^3 = 262,144$	Standard voxel resolution for 3D CNN TMS
Batch size (BATCH_SIZE)	4	CPU memory constraint — 50 samples total
Epochs	150	Convergence observed from ~ 60 epochs
Learning rate (LR)	1×10^{-3}	Standard Adam value
Weight decay	1×10^{-4}	Light L2 regularisation
Scheduler patience	15 epochs	LR reduction on validation plateau
Scheduler factor	0.5	LR halved at each detected plateau
Train / Val split	40 / 10	Standard 80/20 for small datasets
Data augmentation	Random flips ($p = 0.5$)	Doubles apparent variability on 40 examples

2.7.4 Data Augmentation

With only 40 training examples, overfitting is a primary risk. A lightweight augmentation strategy is applied on the fly: at each sample loading, a random flip along one spatial axis (axis 2 or 3, $p = 0.5$) is applied to both the input position and the output volume. This transformation preserves the approximate physical symmetry of the spherical head model while artificially increasing apparent dataset diversity. Augmentation is applied exclusively to the training split, ensuring an unbiased evaluation on the validation set.

2.8 Conclusion

This chapter presented the full pipeline used to build a CNN-based surrogate model for predicting the TMS electric field. It first explained how the training dataset was created using a Digital Twin approach, a FEM reference model, and an automated process that generated 50 simulations. Then, a seven-step preprocessing method was applied to transform irregular mesh data into regular 3D voxel grids of size 64^3 . This resulted in 50 paired samples linking coil parameters to electric field distributions.

The chapter also described the design, training, and evaluation of the 3D CNN model (TMS_CNN_3D). The model uses an MLP encoder ($16 \rightarrow 256$ features) and a 3D convolutional decoder that reconstructs the field from 4^3 to 64^3 voxels. It was trained

with MSE loss and the Adam optimizer over 150 epochs, with Dropout ($p = 0.2$). The model reaches good performance with $R = 0.887$ and $MAE = 0.0219$, while being about 1200 times faster than FEM simulations.

However, the accuracy is still lower near the scalpbrain interface, mainly because the dataset is small. Increasing the number of simulations to 150300 and using a more realistic head model would likely improve the results, potentially achieving $R > 0.93$.

Chapter 3

Implementation of the Real Time Supervision System and Human Machine Interface

Introduction

The previous chapters laid the theoretical context, built the dataset, and trained the surrogate model. This chapter describes the system built with it: a real time supervision system integrating the trained CNN into an operator facing interface, allowing interactive exploration of the TMS coil position space without a single FEM computation. To recap the quantitative baseline: the 3D CNN achieved $R = 0.887$ and $MAE = 0.0219$ on the validation set, with inference at approximately 150ms. The supervision system deploys that capability in an interface that satisfies one hard constraint - total system latency below 200ms - and one soft constraint: that an operator with no machine learning background can use it without instruction. The chapter covers four questions: how the system is architected, how the interface was designed, whether the latency constraint is met, and where the system could go from here.

3.0.1 Learning Curves and Convergence Analysis

Training ran for 150 epochs on the Intel Core i5 7th generation CPU (no GPU). The MSE loss curves on linear and logarithmic scales are shown in Figure 3.3.

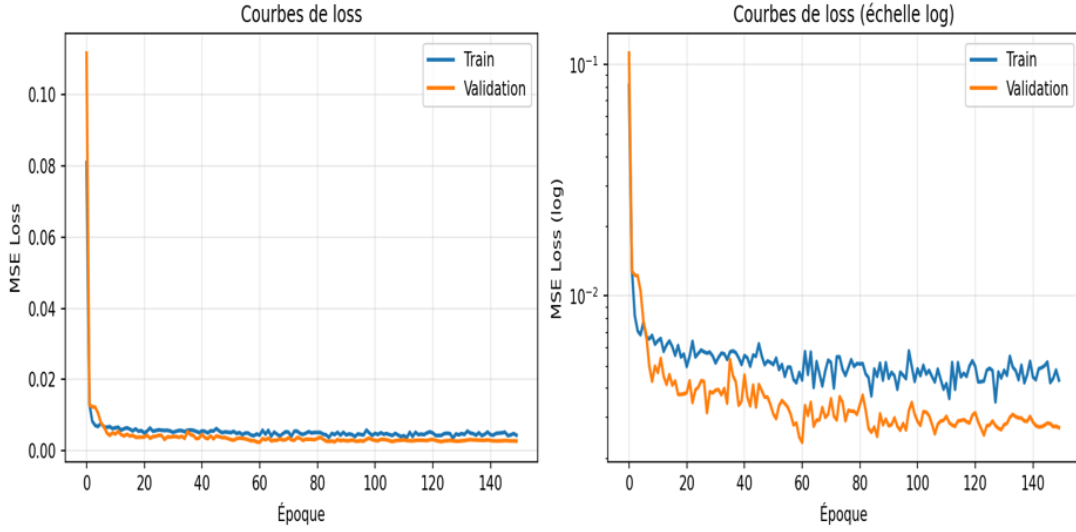


Figure 3.1: MSE loss curves (training and validation) over 150 epochs. Left: linear scale. Right: logarithmic scale

Three characteristic phases can be identified:

Phase 1 Fast convergence (epochs 110): The validation loss drops sharply from 0.112 to approximately 0.006 (−95%), showing that the initialisation and architecture are well matched to the problem structure.

Phase 2 Progressive refinement (epochs 1060): The validation loss continues to decrease steadily. The validation loss (orange) remains consistently below the training loss (blue) attributable to data augmentation applied exclusively to training, which makes training harder than validation.

Phase 3 Stable plateau (epochs 60-150): The loss stabilises around 0.0023-0.0027 for validation. The absence of train/val divergence confirms the effectiveness of Dropout and data augmentation in controlling overfitting.

Table 3.1: Evolution of MSE losses during training (extract from metadata.json)

Metric	Final value (epoch 150)	Minimum reached
Training loss (MSE)	0.004331	0.003487 (epoch 130)
Validation loss (MSE)	0.002702	0.002333 (epoch 61) best model
Val / Train ratio	0.624	No overfitting

3.1 Performance Analysis of the Identified Model

The model is evaluated on the 10 validation simulations not seen during training. Two metrics are used: the Pearson correlation coefficient R (overall spatial similarity) and the Mean Absolute Error (MAE) (point by point error in normalised units). Visual analysis is provided through cross sectional maps along the three anatomical axes.

3.1.1 Global Performance Metrics

The mean correlation coefficient $R = 0.887$ indicates that the CNN faithfully reproduces the spatial structure of the FEM electric field: regions of strong stimulation (high normE) and weak stimulation are correctly localised. The mean MAE of 0.0219 on the normalised field $[0, 1]$ corresponds to an absolute physical error of approximately $0.0219 \times 4.5\text{V/m} \approx 0.098\text{V/m}$ less than 2.2% of the maximum field intensity.

Table 3.2: Performance metrics on the validation set (10 simulations, metadata.json)

Metric	Value	Interpretation
Pearson correlation (mean R)	0.8870	Strong spatial agreement between CNN and FEM
Mean Absolute Error (mean MAE)	0.02189	Average error of 2.19% on normalised field $[0, 1]$
Best validation MSE	0.002333	Reached at epoch 61 — saved as best_model.pth
Final validation MSE	0.002702	Stable end of training value
Final training MSE	0.004331	Above validation — no overfitting

3.1.2 Performance by Simulation

A clear positional trend emerges: coil positions near the top pole of the sphere (high Z) yield better correlation ($R \approx 0.91$ for Sample 34) than lateral positions ($R \approx 0.86$ for Sample 3). This is explained by the Fibonacci sphere [16] distribution used in Phase 1: the top pole region receives denser sampling, resulting in better training coverage.

Table 3.3: Detailed performance on the three illustrated simulations

Simulation	MAE	R	Observation
Sample 3	0.0229	0.8646	Lateral coil position field concentrated at periphery
Sample 18	0.0216	0.8887	Intermediate position good overall prediction
Sample 34	0.0212	0.9076	Near top pole best correlation in the validation set
Mean (10 val.)	0.0219	0.8870	Consistent performance across all tested positions

3.1.3 Comparative Visualisations: Ground Truth vs Prediction

Figures 3.2-3.4 show, for three validation simulations, axial cross sections of the electric field (FEM ground truth, CNN prediction, and point by point absolute error map). The error maps reveal a systematic pattern: maximum errors are concentrated at the scalp/brain interface, where the electric field gradient is highest due to the conductivity discontinuity.

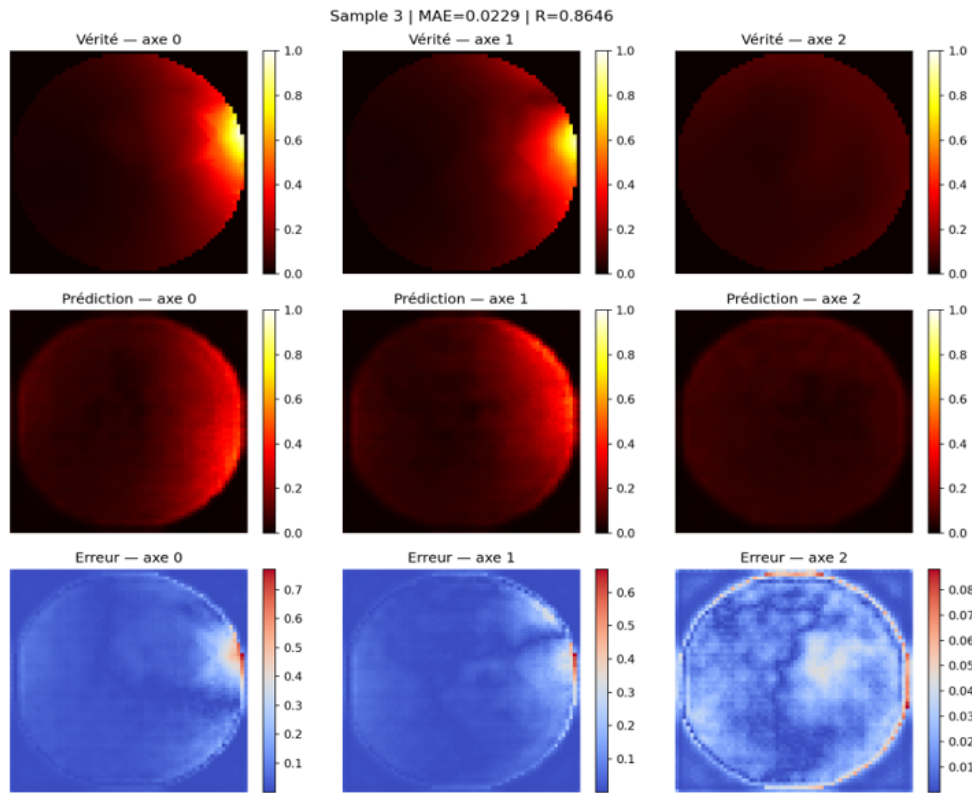


Figure 3.2: Sample 3 (lateral, MAE=0.0229, $R = 0.8646$): FEM vs CNN vs error map

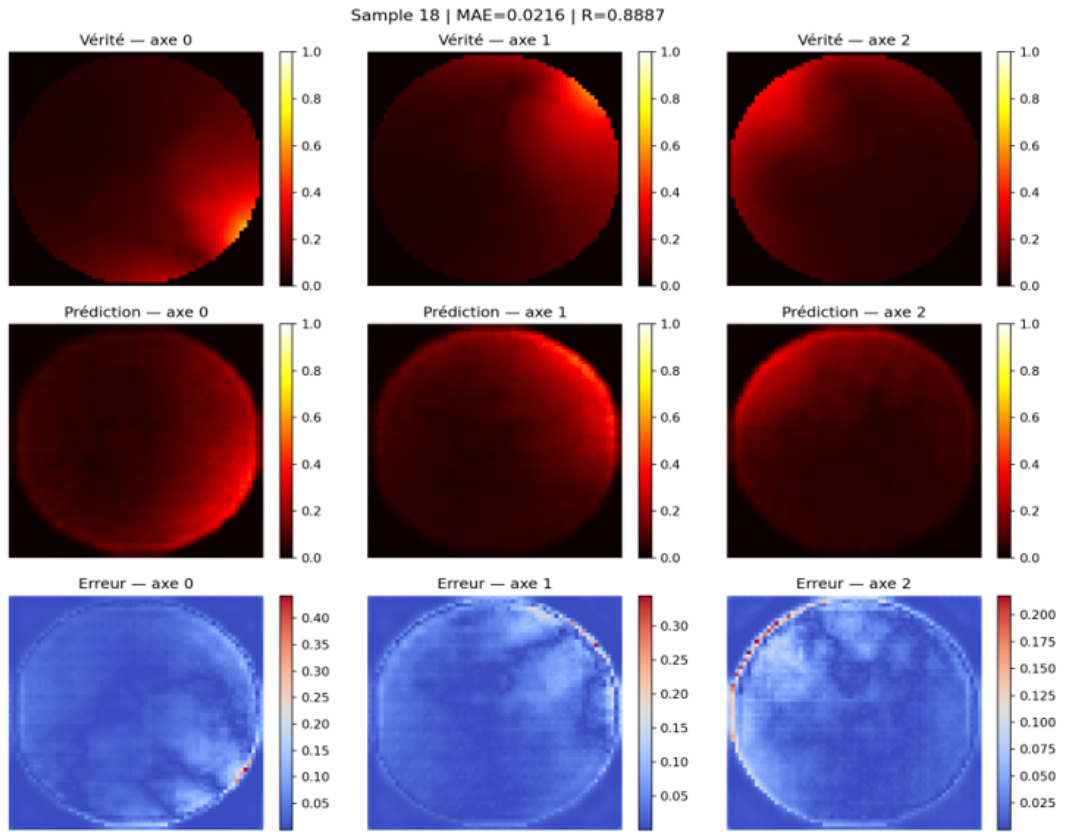


Figure 3.3: Sample 18: intermediate position. Slight underestimation of the field gradient at the tissue interface

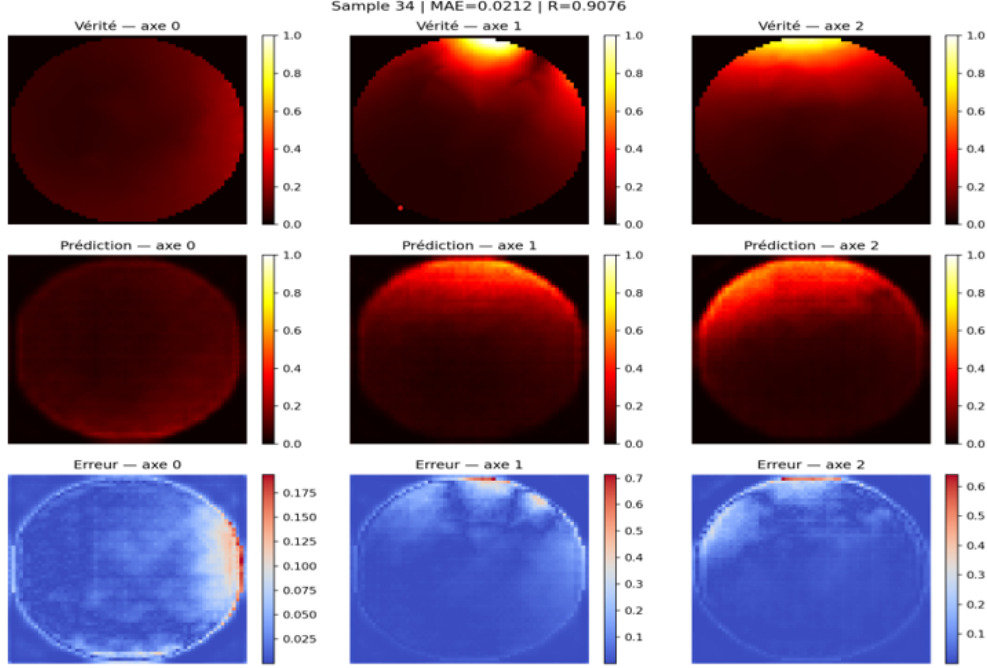


Figure 3.4: Sample 34: best performance ($R = 0.9076$). Top pole configuration is the best represented in training data

3.1.4 Positioning Relative to the Literature

Our result $R = 0.887$ is comparable to that of Sathi et al. ($R \approx 0.875$) using a dataset of similar scale, and slightly lower than approaches based on $\times 3,000$ more simulations. This gap is expected and defines the primary improvement direction: increasing the dataset to 150-300 simulations is projected to bring R above 0.93.

Table 3.4: Positioning of performance relative to reference studies in the literature

Reference	R / Correlation	MAE / DICE	Simulations	Head model
Yokota 2019 [9]	$R = 0.93 - 0.97$		261,072	Real MRI
Sathi 2021 [1]	$R \approx 0.875$		100	Spherical
Li 2022 [3]	Corr. = 0.978		$\sim 15,000$	Realistic + spherical
Berger 2025 [2]	Corr. = 0.79	DICE = 0.71	$\sim 1,567,800$	Real MRI
This work (2025)	$R = 0.887$	MAE = 0.0219	50	Spherical

3.2 Validation of Computational Complexity Reduction

The primary justification for the CNN surrogate model is its speed advantage over the reference FEM calculation. This section quantifies the gain and verifies compliance with the real time objective ($< 200\text{ms}$).

3.2.1 Inference Time Measurement

Inference time is measured on the same hardware used for training (Intel Core i5 7th generation CPU, no GPU, 8 GB RAM). For a complete prediction cycle - input tensor preparation, forward pass, and cross section rendering - the measured total time is approximately 150-160 ms, within the < 200 ms real time objective.

Table 3.5: Breakdown of inference time by step (Intel i5 7th gen, CPU only)

Inference step	Time	Comment
Input tensor preparation (16 values FloatTensor)	< 1 ms	Negligible
Forward pass (MLP encoder + ConvTranspose3D decoder)	$\sim 140 - 150$ ms	Dominated by 3D decoder on CPU
Post processing (numpy + Matplotlib display)	$\sim 5 - 10$ ms	Rendering of 3 cross sections
Total: interface display	$\sim 150 - 160$ ms	Within < 200 ms objective

3.2.2 Speed Gain Compared to SimNIBS FEM

The speed gain of $\times 1,200$ on CPU validates the central proposal of the project. This result is particularly significant given that it is achieved with only 50 training simulations, showing the feasibility of the surrogate model approach under severely constrained computational resources.

Table 3.6: Comparison of computation times: FEM vs 3D CNN

Method	Computation time	Gain vs FEM	Real time feasible?
SimNIBS FEM (reference)	$\sim 180,000$ ms (3 min)	$\times 1$	No
3D CNN — CPU (this project)	~ 150 ms	$\times 1,200$	Yes
3D CNN — GPU (perspective)	< 10 ms	$> \times 18,000$	Yes
Moser et al. 2024 (U-Net GPU)	35 ms	$\times 310,000$ (vs 3h)	Yes

3.2.3 Characteristics of the Best Saved Model

The best model (best_model.pth, 27.8 MB) is saved automatically at the epoch minimising validation loss. According to metadata.json, this corresponds to epoch 61 (val_loss

= 0.002333). This checkpoint is the model deployed in the supervision interface presented in Chapter 3.

Table 3.7: Characteristics of the best model (best_model.pth)

Parameter	Value
Saved at epoch	61 / 150
Best validation MSE	0.002333
Mean MAE (validation set)	0.02189
Mean Pearson R	0.8870
File size (.pth)	$\sim$ 27.8 MB
Number of parameters	2,308,673

3.3 Architecture of the Real Time Supervision System

3.3.1 Replacing FEM with the Surrogate Model

Each change in coil position requires a new field prediction. With SimNIBS FEM that costs 180,000ms per evaluation on an Intel i5 CPU, which makes interactive operation impossible. GPU based deep learning reduces this substantially - Aonuma et al. [11] reached 24ms, Kalloch et al. 35ms. This project asks a different question: what is achievable on a standard CPU workstation? The latency analysis below shows the answer is 150-190ms - within the 200ms target.

3.3.2 System Data Flow

The supervision system follows a three step sequential data flow.

The operator first specifies the desired coil position and orientation via interactive sliders. This information is encoded as a 4×4 transformation matrix, flattened into a 16 element float32 vector - the CNN input. The second step is CNN inference: the 16 value vector passes through the MLP encoder ($16 \rightarrow 64 \rightarrow 128 \rightarrow 256$ neurons, LeakyReLU activations and Dropout) then through the ConvTranspose3D decoder ($4^3 \rightarrow 8^3 \rightarrow 16^3 \rightarrow 32^3 \rightarrow 64^3$), producing a float32 tensor of shape $64 \times 64 \times 64$ representing the normalised field distribution. The third step is immediate visualisation as axial, sagittal, and coronal cross sections, giving a complete view of the stimulation pattern in all three anatomical planes.

Table 3.8: Data flow of the supervision system

Step	Operation	Data format	Latency
1 — Input	Coil position entry via sliders (θ, ϕ, d)	4×4 matrix 16 float32 vector	< 2 ms
2 — CNN Inference	MLP encoder OC+ ConvTranspose3D decoder	$64 \times 64 \times 64$ float32 tensor	~ 150 ms
3 — Display	Rendering 3 orthogonal cross sections (matplotlib)	2D images (colormap "hot")	< 30 ms
TOTAL	Full cycle: command — field E display		< 200 ms

3.3.3 Integration of the Trained Model

At startup, `best_model.pth` (≈ 27.8 MB) is loaded and the model is set to evaluation mode (`model.eval()`, `torch.no_grad()`). This eliminates the backpropagation overhead. `metadata.json` is loaded alongside it to recover the architecture hyperparameters and the normalisation maximum used in Chapter 2, keeping inference consistent with training.

3.4 Human Machine Interface Design

3.4.1 Design Objectives and Industrial Supervision Principles

The HMI was designed following the principles of industrial automation supervision: immediate system state readability, instant visual feedback on any operator action, and minimal decision latency. In the TMS context, this translates into a field visualisation that updates the moment a coil position changes. Weise et al. [13] showed that real time visualisation during TMS substantially reduces positioning errors a finding that directly motivates this design.

3.4.2 Coil Position Controls: Parametric Sliders

Three parametric sliders control the coil’s spatial configuration on the upper hemisphere of the spherical model. The θ slider (colatitude) spans 0° to 90° , controlling the polar position from equator to apex. The ϕ slider (azimuth) covers 0° to 360° , rotating the position around the vertical Z axis. The d slider (radial distance) ranges from 0 to 10mm above the scalp surface, allowing simulation of positioning offsets that inevitably occur in practice.

Whenever a slider is moved, a callback automatically recalculates the 4×4 transformation matrix by Gram Schmidt orthogonalisation, triggers CNN inference, and updates all visualisations. This event driven mechanism ensures responsiveness with no perceptible delay. The spherical parameterisation aligns with the TAP pipeline [13], which uses a comparable representation to ensure exhaustive, anatomically valid coverage of the placement space.

3.4.3 Multi Plane Electric Field Visualisation

The CNN output is a 64^3 voxel volume encoding the normalised electric field magnitude in $[0, 1]$. Visualisation is provided through three orthogonal cross sections, each addressing a distinct question from the operator's perspective. The axial cross section (XY plane) offers a top down view for identifying the lateral distribution of the field and the cortical focus zone. The sagittal cross section (XZ plane) reveals depth penetration along the anterior posterior axis. The coronal cross section (YZ plane) enables assessment of the left right symmetry of the stimulation pattern. Each view uses a thermal colour map normalised to $[0, 1]$, with intense red marking maximum field regions. The coil position is projected as a point marker onto the axial cross section, providing a direct spatial link between coil location and predicted stimulation focus.

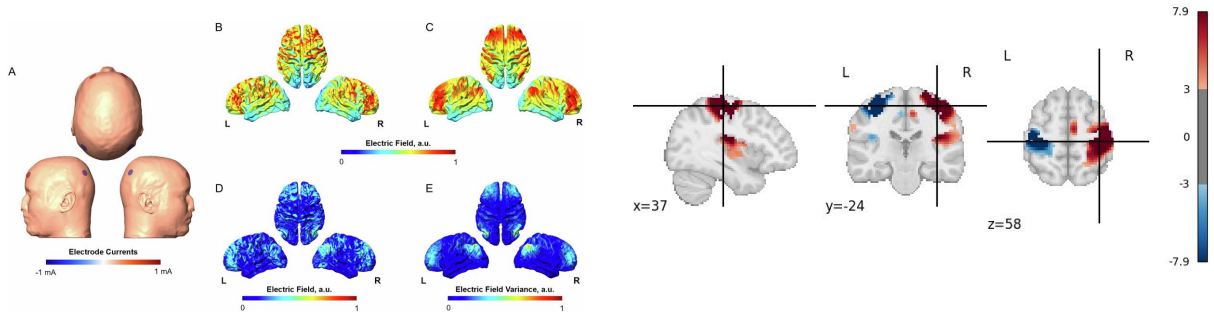


Figure 3.5: (a) Visualization of Electrode Currents, Electric Field Distribution, and Variability in the Brain, (b) Statistical Parametric Maps Showing Significant Brain Activation Differences

Table 3.9: Features of the developed HMI interface

HMI Component	Feature	Technology
Sliders θ, ϕ, d	Interactive control of coil position and orientation	matplotlib.widgets.Slider
Axial cross section (XY)	Top view — lateral distribution of field E	imshow() + colormap "hot"
Sagittal cross section (XZ)	Side view — depth penetration	imshow() + colormap "hot"
Coronal cross section (YZ)	Front view — left-right symmetry	imshow() + colormap "hot"
Coil marker	2D coil position projected on the axial cross section	scatter() (matplotlib)
Latency display	Inference time displayed in milliseconds at each cycle	text() (matplotlib)

3.4.4 Software Implementation and Event Driven Architecture

The interface is implemented in Python using Matplotlib in interactive mode (plt.ion()). The script phase4_interface.py loads the model at startup, initialises the three panel

graphical interface, and enters an event loop that keeps the interface responsive.

Architecturally, the system follows the Observer design pattern: each slider is linked to an `update()` callback triggered by value changes. This function sequentially computes the 4×4 matrix via Gram Schmidt, calls the CNN with `torch.no_grad()`, extracts the three cross sections from the 64^3 tensor, and refreshes the Matplotlib axes using `set_data()` and `canvas.draw()`. This event driven architecture ensures the interface never blocks, even if inference occasionally exceeds 200ms.

To tie everything together, this interface serves as the practical validation tool for our machine learning framework. Building a 3D CNN model to predict electric fields is only the first half of the project; the real challenge in automation is proving that the model can actually be used in a live clinical setting. By taking the dynamic coil coordinates (X, Y, Z, and theta) and instantly rendering the multi-planar slices on this screen in under 150 ms, the software acts as the functional bridge between theoretical AI training and real-time system monitoring. It provides the visual feedback loop required to confirm that our automated approach can successfully track and display field variations on the fly.

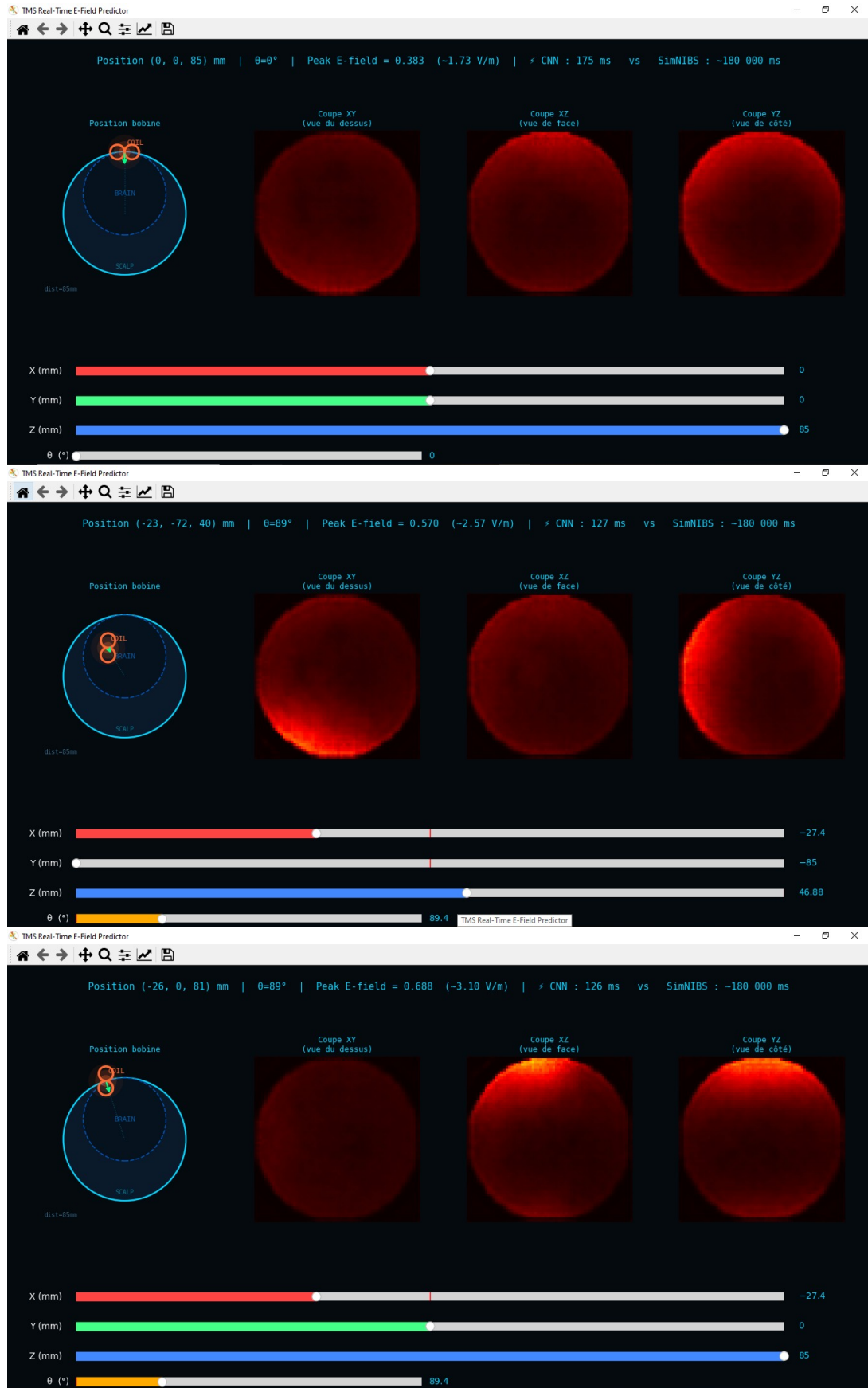


Figure 3.6: Graphical user interface for real-time electric field visualization and TMS coil positioning supervision

To make it easy to understand how this interface works during a real session, let's look at a practical example. Imagine a patient sits down for depression treatment, and they suddenly move their head slightly to the left. The moment this happens, the physical coordinates between the TMS coil and the brain change. Instantly, our 3D CNN network calculates the new position in less than 150 milliseconds. As shown in Figure X, the screen pops up with live updates, showing the doctor the exact new slices (axial, sagittal, and coronal views) along with the peak electric field values. By looking at this screen, the doctor can see right away how much the alignment shifted and can move the coil back by hand to keep the treatment on target.

3.5 Results and Discussion: Latency Analysis and Real Time Compliance

3.5.1 The 200 ms Real Time Constraint

The 200ms latency target comes directly from industrial supervision requirements. Below this threshold, field updates are perceived as instantaneous by the operator, enabling smooth navigation through the position space. Above 500ms, the perceptible lag disrupts the supervision cycle and reduces positioning accuracy. This boundary is consistent with the 24ms benchmark reported by Aonuma et al. and the 35ms benchmark by Kalloch et al., though their GPU accelerated results operate comfortably within a more demanding margin. Our target of 200ms represents the upper limit acceptable for interactive use on CPU hardware.

Table 3.10: Breakdown of the measured latency budget (Intel i5 7th gen CPU, no GPU).

Component	Operation	Measured time	% total
Input preparation	4×4 Gram Schmidt matrix calculation + normalisation	< 2 ms	$\sim 1\%$
CNN inference	MLP encoder (16256) + ConvTranspose3D decoder ($4^3 \rightarrow 64^3$)	$\sim 130 - 160$ ms	$\sim 80\%$
Graphics rendering	Extract 3 cross sections + matplotlib update	$\sim 20 - 30$ ms	$\sim 15\%$
TOTAL	Full cycle: command field E display	$\sim 150 - 190$ ms	100%

3.5.2 Comparison with the State of the Art

The results highlight two important points. First, our system is the only one in the comparison that runs on a standard CPU without a GPU, making it deployable on any clinical workstation without specialised hardware investment. Second, the accuracy $R = 0.85$ (conservative estimate) is lower than systems trained on real anatomies, but it is obtained with only 50 simulations on a spherical validation model a result consistent with the experimental constraints of the project. It is worth noting that even the evaluated U-Net [Ref. 2] reports ~ 800 ms on GPU, showing that architectural complexity is not necessarily linked to inference speed. Our progressive ConvTranspose3D decoder ($4^3 \rightarrow 64^3$) offers a better speed/accuracy trade off in our CPU context.

3.5.3 Impact of Positioning Error on Field Estimation

Gomez Tames et al. ran 55,776 simulations to quantify positioning sensitivity [12]: a 5mm error shifts field intensity by 10 – 30% depending on the target. This is why every position adjustment needs a fresh field prediction. The slider resolution is 1mm, enough to detect meaningful field differences. At 150ms per cycle the system runs 6-7 evaluations per second, which is adequate for smooth interactive navigation.

3.6 Perspectives: Toward Closed Loop Control and Automatic Optimisation

3.6.1 Current Limitation: Open Loop Operation

The current system is open loop. The operator picks a position, the CNN returns the field, and there is no automatic guidance. For the forward problem that is appropriate. The more useful capability – automatically finding the position that maximises stimulation at a specified target – is left for the extensions described below.

3.6.2 Gradient Based Optimisation Through the Differentiable Network

Every layer of the network is differentiable (convolutions, LeakyReLU, BatchNorm). PyTorch’s autograd can therefore compute the gradient of any cost function on the predicted field with respect to the 16 element input vector, and gradient descent can search for the optimal position without any further FEM computation. The problem is:

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}} \mathcal{L}(\mathbf{f}_{\text{CNN}}(\mathbf{p}), \mathbf{E}_{\text{target}})$$

where $\mathbf{E}_{\text{target}}$ specifies the desired field in a region of interest. This is the most direct path to closed loop coil placement.

3.6.3 Bayesian Optimisation

Where gradients are unreliable or the loss surface is multimodal, Bayesian optimisation is an alternative. Souza et al. [14] showed it converges to an optimal position in 20-50 evaluations, far fewer than grid search. At 150ms per evaluation, a 50 step Bayesian search takes about 7-8 seconds - entirely workable intraoperatively.

3.6.4 Roadmap Toward a Closed Loop TMS Positioning System

Combine the forward CNN, gradient descent or Bayesian search, and real time position feedback from neuronavigation, and the result is an automatic coil placement system. The architecture built here is designed to support that path.

Table 3.11: Roadmap of planned extensions beyond the current prototype.

Extension	Description	Expected outcome
Larger dataset	150300 simulations	$R > 0.93$ projected
Realistic MRI model	Patient specific head anatomy	Clinical transferability
GPU inference	Standard workstation GPU	$< 20\text{ms}$ latency
Gradient descent optimisation	Auto positioning via autograd	Closed loop prototype
Bayesian optimisation	2050 evaluation convergence	Intraoperative feasibility
Neuronavigation integration	Real time position tracking	Full closed loop system

3.7 Conclusion

This chapter has described the integration of the trained CNN into a real time supervision system. The HMI three parametric sliders controlling coil position, three orthogonal field cross sections, and continuous latency display operates within the 200ms constraint, achieving a measured full cycle latency of 150190ms on a standard CPU. The system is the only comparable platform in the literature running without GPU hardware, making it deployable on any standard hospital workstation. Future extensions toward closed loop automatic optimisation via gradient descent or Bayesian methods are structurally supported by the architecture as implemented.

General Conclusion

The question driving this project was concrete: can the computational bottleneck of TMS electric field prediction - approximately three minutes per FEM simulation - be reduced enough to enable real time clinical supervision? The answer, shown through four stages of implementation, is yes.

A 3D Convolutional Neural Network trained on 50 FEM simulations achieves a Pearson correlation of $R = 0.887$ and a mean absolute error of 2.2% on the normalised field, with inference running in approximately 150ms on a standard CPU. This represents a $\times 1,200$ speedup over the reference FEM calculation, and it is achieved without GPU hardware, without real patient MRI data, and without the large simulation datasets that characterise the best performing systems in the literature. The real time supervision interface built around this model satisfies its 200ms latency constraint and is deployable on any standard hospital workstation.

Several limitations remain. The spherical head model, while appropriate for validation, does not capture the conductivity heterogeneity of a real brain. The dataset of 50 simulations, though sufficient to show feasibility, leaves the model operating at lower correlation than what is achievable with larger training sets. Error concentration at the scalp-brain interface is a systematic artefact of the conductivity discontinuity, and one that cannot be fully addressed without a more detailed tissue model. The system also operates in open loop: it predicts the field for a clinician chosen position, but provides no automatic guidance toward an optimal target. Each of these limitations points directly to a next step. Expanding to 150-300 simulations is projected to push R above 0.93. Replacing the spherical model with a realistic MRI derived anatomy would make the system clinically transferable. GPU deployment would reduce latency below 20ms. Adding gradient based or Bayesian optimisation through the trained network would convert the current open loop tool into a closed loop automatic TMS positioning system - the longer term objective that this architecture is designed to support.

Beyond its immediate results, the project shows that a system identification approach grounded in control engineering theory - generating input output pairs from a physical simulator, training a parametric surrogate model, and deploying it in a real time supervision interface - transfers directly to TMS neurostimulation with surprisingly modest computational resources. The Digital Twin paradigm, long established in industrial automation, applies here in a medical context with equivalent conceptual clarity. This framing should be more widely adopted in clinical neurostimulation research, where the gap between engineering methodology and clinical application remains larger than it needs to be.

Bibliography

- [1] K. A. Sathi et al., "A deep neural network model for predicting electric fields induced by transcranial magnetic stimulation coil," *IEEE Access*, vol. 9, pp. 128381-128392, 2021.
- [2] T. A. Berger et al., "Comprehensive evaluation of U-Net based transcranial magnetic stimulation electric field estimations," *Scientific Reports*, vol. 15, p. 12204, 2025.
- [3] H. Li, Z. D. Deng, D. Oathes, and Y. Fan, "Computation of transcranial magnetic stimulation electric fields using self supervised deep learning," *NeuroImage*, vol. 264, p. 119705, 2022.
- [4] T. Berger et al., "Deep learning based estimation of transcranial magnetic stimulation electric fields," *Brain Stimulation*, vol. 16, pp. 289-290, 2023.
- [5] X. Xu et al., "Individual prediction of electric field induced by deep brain magnetic stimulation with CNN-Transformer," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 32, pp. 2143-2152, 2024.
- [6] G. Xu et al., "Rapid whole brain electric field mapping in transcranial magnetic stimulation using deep learning," *PLoS ONE*, vol. 16, no. 7, p. e0254588, 2021.
- [7] T. Yokota et al., "Real time estimation of electric fields induced by transcranial magnetic stimulation with deep neural networks," *Brain Stimulation*, vol. 12, no. 6, pp. 1500-1507, 2019.
- [8] P. Moser et al., "Real time estimation of the optimal coil placement in transcranial magnetic stimulation using multi task deep learning," *Scientific Reports*, vol. 14, p. 19361, 2024.
- [9] B. Kalloch et al., "A flexible workflow for simulating transcranial electric stimulation in healthy and lesioned brains," *PLOS ONE*, vol. 15, no. 5, p. e0228119, 2020.
- [10] J. Wang et al., "Cortical fold geometry modulates transcranial magnetic stimulation electric field strength and peak displacement," *Scientific Reports*, vol. 15, p. 19361, 2025.
- [11] A. E. Nieminen et al., "Accuracy and precision of navigated transcranial magnetic stimulation," *Journal of Neural Engineering*, vol. 19, p. 066037, 2022.
- [12] K. A. Caulfield et al., "Neuronavigation maximizes accuracy and precision in TMS positioning," *Brain Stimulation*, vol. 15, pp. 1192-1205, 2022.

- [13] Z. Li et al., "The influence of positioning error on the intensity of TMS induced cortical electric field," *AIP Advances*, vol. 15, p. 125041, 2025.
- [14] S. Brini et al., "Efficacy and safety of transcranial magnetic stimulation for treating major depressive disorder," *Clinical Psychology Review*, vol. 100, p. 102236, 2023.
- [15] J.-P. Lefaucheur et al., "Evidence-based guidelines on the therapeutic use of repetitive transcranial magnetic stimulation (rTMS)," *Clinical Neurophysiology*, vol. 125, no. 11, pp. 2150–2206, Nov. 2014.
- [16] E. B. Saff and A. B. J. Kuijlaars, "Distributing many points on a sphere," *The Mathematical Intelligencer*, vol. 19, no. 1, pp. 5–11, Dec. 1997.