



Titre: Green Function and Electromagnetic Potential for Computer Vision
Title: and Convolutional Neural Network Applications

Auteur: Dominique Beaini
Author:

Date: 2019

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Beaini, D. (2019). Green Function and Electromagnetic Potential for Computer
Citation: Vision and Convolutional Neural Network Applications [Ph.D. thesis, Polytechnique
Montréal]. PolyPublie. <https://publications.polymtl.ca/3982/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/3982/>
PolyPublie URL:

**Directeurs de
recherche:** Sofiane Achiche, & Maxime Raison
Advisors:

Programme: Génie mécanique
Program:

UNIVERSITÉ DE MONTRÉAL

GREEN FUNCTION AND ELECTROMAGNETIC POTENTIAL FOR COMPUTER VISION
AND CONVOLUTIONAL NEURAL NETWORK APPLICATIONS

DOMINIQUE BEAINI

DÉPARTEMENT DE GÉNIE MÉCANIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIAE DOCTOR
(GÉNIE MÉCANIQUE)

JUIN 2019

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse est intitulée :

GREEN FUNCTION AND ELECTROMAGNETIC POTENTIAL FOR COMPUTER VISION
AND CONVOLUTIONAL NEURAL NETWORK APPLICATIONS

présentée par : BEAINI Dominique

en vue de l'obtention du diplôme de : Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de :

Mme CHERIET Farida, Ph. D., présidente

M. ACHICHE Sofiane, Ph. D., membre et directeur de recherche

M. RAISON Maxime, Ph. D., membre et codirecteur de recherche

M. PAMPLONA SEGUNDO Mauricio, D. Sc., membre

M. REBOUCAS DE OLIVEIRA Luciano, D. Sc., membre externe

ACKNOWLEDGEMENTS

The first acknowledgment for my Ph.D. thesis goes to my directors Prof. Sofiane Achiche and Prof. Maxime Raison for their continued support, patience and motivation. They are both wise, knowledgeable, kind and helpful. It is hard to imagine going this far in my Ph.D. without their support, their open mind and their advice.

I would like to add a special thanks to the newest member of my family, my wife Nazanin Naderi, whom I married in the middle of my Ph.D. studies. Her continued kindness and support allowed me to outperform myself by starting each of my work days with happiness. She always pushed me to overcome every difficulty and never give up, and for that, I thank her immensely.

Other special thanks go to my parents for always being there for me and pushing me to overcome myself since my youth. My older sister Véronique for guiding me through life and providing me with an example of perseverance and kindness. My younger brother Rodrigue for being my lifelong rival and for his continued support at Polytechnique. I wish him good luck for his current Master's degree and (maybe) his Ph.D.

I also acknowledge all of my co-workers for their support, their positive attitude and the great working environment that they provided. In truth, I considered each of my co-worker to be my friend, since they only provided joy and fun to the lab, and I hope they write their thesis soon!

I would like to name Mahdis Asaadi, Olivier Brochu-Dufour, and Alexandre Duperré, who worked with me as highly dedicated interns. They helped me a lot in the progress of my Ph.D., but also taught me a lot from their own perspective and experience. I also want to thank Abolfazl Mohebbi for deciding to continue the project for industrial applications.

Another thank you goes to my friend and my favorite philosophy partner Fabrice Nonez. Although his true passion is pure mathematics, he was always there to help me have a deeper understanding of the applied mathematics that I was developing for my Ph.D.

Finally, I would like to thank all my friends and family for their continuous support. I can finally answer them when they ask me the everyday question, “when do you finish your Ph.D.?”

RÉSUMÉ

Pour les problèmes de vision machine (CV) avancées, tels que la classification, la segmentation de scènes et la détection d'objets saillants, il est nécessaire d'extraire le plus de caractéristiques possibles des images. Un des outils les plus utilisés pour l'extraction de caractéristiques est l'utilisation d'un noyau de convolution, où chacun des noyaux est spécialisé pour l'extraction d'une caractéristique donnée. Ceci a mené au développement récent des réseaux de neurones convolutionnels (CNN) qui permet d'optimiser des milliers de noyaux à la fois, faisant du CNN la norme pour l'analyse d'images. Toutefois, une limitation importante du CNN est que les noyaux sont petits (généralement de taille 3×3 à 7×7), ce qui limite l'interaction longue-distance des caractéristiques. Une autre limitation est que la fusion des caractéristiques se fait par des additions pondérées et des opérations de mise en commun (moyennes et maximums locaux). En effet, ces opérations ne permettent pas de fusionner des caractéristiques du domaine spatial avec des caractéristiques puisque ces caractéristiques occupent des positions éloignées sur l'image.

L'objectif de cette thèse est de développer des nouveaux noyaux de convolutions basés sur l'électromagnétisme (EM) et les fonctions de Green (GF) pour être utilisés dans des applications de vision machine (CV) et dans des réseaux de neurones convolutionnels (CNN). Ces nouveaux noyaux sont au moins aussi grands que l'image. Ils évitent donc plusieurs des limitations des CNN standards puisqu'ils permettent l'interaction longue-distance entre les pixels de l'image. De plus, ils permettent de fusionner les caractéristiques du domaine spatial avec les caractéristiques du domaine du gradient. Aussi, étant donné tout champ vectoriel, les nouveaux noyaux permettent de trouver le champ vectoriel conservatif le plus rapproché du champ initial, ce qui signifie que le nouveau champ devient lisse, irrotationnel et conservatif (intégrable par intégrale curviligne).

Pour répondre à cet objectif, nous avons d'abord développé des noyaux convolutionnels symétriques et asymétriques basés sur les propriétés des EM et des GF et résultant en des noyaux qui sont invariants en résolution et en rotation. Ensuite, nous avons développé la première méthode qui permet de déterminer la probabilité d'inclusion dans des contours partiels, permettant donc d'extrapoler des contours fins en des régions continues couvrant l'espace 2D. De plus, la présente thèse démontre que les noyaux basés sur les GF sont les solveurs optimaux du gradient et du Laplacien. De ce fait, même s'il n'existe pas de solution exacte au gradient et au Laplacien, les

noyaux développés trouvent la solution la plus rapprochée possible d'un résultat, et ce en étant au moins 3.2 fois plus rapide que toute autre méthode de la littérature.

Ainsi, en utilisant notre solveur de gradient, nous avons développé la première méthode qui permet de combiner directement des matrices de contours avec des matrices de salience. L'amélioration des matrices de salience est en moyenne 6.6 fois supérieure au plus proche compétiteur sur des bases de données sélectionnées. Ensuite, pour améliorer notre algorithme de salience, nous avons développé le modèle DSS-GIS qui combine les contours et à la salience directement à l'intérieur d'un CNN profond. Cette combinaison a permis d'améliorer la performance du CNN, de réduire le surapprentissage et de réduire le temps d'apprentissage, pour une augmentation de seulement 10% du temps d'exécution. En plus, la couche GIS a permis d'améliorer les performances du *F-measure* de 3.9% dans le cas d'images bruitées et de 2.3% dans le cas d'images à faible luminosité. Finalement, nous avons développé un premier prototype qui permet d'utiliser les GF à différentes profondeurs dans un réseau de classification de chiffres. Ce prototype fonctionne en transformant le champ vectoriel de caractéristiques en un champ conservatif. Les premiers résultats sont prometteurs, car ils montrent une réduction du temps d'entraînement d'un facteur 5.2, une réduction du bruit dans les courbes d'apprentissage et une réduction de 28% de l'erreur de classification.

La principale retombée scientifique de la présente thèse est la création d'une nouvelle catégorie d'opérations pouvant être utilisés dans les CNNs. Ces opérations basées sur les GF permettent aux CNN de combiner l'information du domaine de l'image avec l'information du domaine du gradient, ce qui diffèrent entièrement des autres catégories d'opérations, soit les noyaux de convolutions, la réduction de taille (pooling) et les fonctions d'activations. Les GF permettent au CNN d'avoir un champ réceptif illimité, et ce à tout emplacement dans le réseau. De plus, ils permettent de convertir en un champ conservatif tout champ d'informations contenus dans les CNN. Enfin, dans le but d'étendre la portée du travail, ces opérations ont été codées dans différents langages, soit Matlab, C++ (OpenCV) et Python (Tensorflow et Pytorch).

ABSTRACT

For advanced computer vision (CV) tasks such as classification, scene segmentation, and salient object detection, extracting features from images is mandatory. One of the most used tools for feature extraction is the convolutional kernel, with each kernel being specialized for specific feature detection. In recent years, the convolutional neural network (CNN) became the standard method of feature detection since it allowed to optimize thousands of kernels at the same time. However, a limitation of the CNN is that all the kernels are small (usually between 3×3 and 7×7), which limits the receptive field. Another limitation is that feature merging is done via weighted additions and pooling, which cannot be used to merge spatial-domain features with gradient-domain features since they are not located at the same pixel coordinate.

The objective of this thesis is to develop electromagnetic (EM) convolutions and Green's functions (GF) convolutions to be used in Computer Vision and convolutional neural networks (CNN). These new kernels do not have the limitations of the standard CNN kernels since they allow an unlimited receptive field and interaction between any pixel in the image by using kernels bigger than the image. They allow merging spatial domain features with gradient domain features by integrating any vector field. Additionally, they can transform any vector field of features into its least-error conservative field, meaning that the field of features becomes smooth, irrotational and conservative (line-integrable).

At first, we developed different symmetrical and asymmetrical convolutional kernel based on EM and GF that are both resolution and rotation invariant. Then we developed the first method of determining the probability of being inside partial edges, which allow extrapolating thin edge features into the full 2D space. Furthermore, the current thesis proves that GF kernels are the least-error gradient and Laplacian solvers, and they are empirically demonstrated to be faster than the fastest competing method and easier to implement.

Consequently, using the fast gradient solver, we developed the first method that directly combines edges with saliency maps in the gradient domain, then solves the gradient to go back to the saliency domain. The improvement of the saliency maps over the F-measure is on average 6.6 times better than the nearest competing algorithm on a selected dataset. Then, to improve the saliency maps further, we developed the DSS-GIS model which combines edges with salient regions deep inside the network. This combination helped improve the performance and reduce the overfitting of the

model using a single GF-based kernel at the last layer of each branch. The added GIS layer allowed an average F-measure improvement of 3.9% for noisy images and 2.3% for low-light images with only 10ms of additional computation cost. Finally, we developed an early prototype that uses the GF convolution at different points inside a classification network for digit recognition. It acts by transforming the field of features into the nearest possible conservative field. Early results show that it helped reduce the training time by a factor 5, reduce the noise in the validation curve and reduce the testing error by 28%, without increasing the computational capacity of the network.

The main outcome of the current thesis is the creation of GF-based operations, a novel category of operations that can be used to improve CNN's. Standard operations used in CNN are the convolutions, the pooling and the activation functions. The GF-based operations do not fit in any of these categories as they offer completely novel properties, allowing the network to have an unlimited receptive field at any given layer, to operate in the gradient-domain and to convert its features into conservative and physically interpretable features. Furthermore, the GF-based operations were written into different languages: Matlab, C++ (OpenCV) and Python (Tensorflow and Pytorch); allowing to deliver the work to the computer vision and machine learning community.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	III
RÉSUMÉ.....	IV
ABSTRACT	VI
TABLE OF CONTENTS	VIII
LIST OF TABLES	XIV
LIST OF FIGURES.....	XV
LIST OF SYMBOLS AND ABBREVIATIONS.....	XXIII
LIST OF APPENDICES	XXIX
CHAPTER 1 INTRODUCTION.....	1
1.1 Objectives.....	1
1.2 Methodology	2
1.3 Overview of the thesis.....	4
CHAPTER 2 LITERATURE REVIEW	6
2.1 Nature-based algorithms	6
2.1.1 Biology inspired techniques	6
2.1.2 Physics-inspired techniques	6
2.2 Shape and partial contour analysis	7
2.2.1 Electromagnetic convolutions	7
2.2.2 Space probability analysis	8
2.3 Gradient-domain image editing.....	9
2.3.1 Poisson solver for seamless blending.....	9
2.3.2 Green's function	9
2.4 Feature detection	10

2.4.1	Edge detection	10
2.4.2	Salient object detection	11
2.4.3	Saliency enhancement	12
2.5	The convolutional neural network.....	13
2.5.1	Convolutional neural networks	13
2.5.2	CNN architectures for feature detection.....	14
2.5.3	Green's function in neural networks	17
2.6	Summary of the problematic	17
CHAPTER 3 ARTICLE 1: NOVEL CONVOLUTION KERNELS FOR COMPUTER VISION AND SHAPE ANALYSIS BASED ON ELECTROMAGNETISM		18
3.1	Introduction	19
3.1.1	Related Work.....	19
3.1.2	Proposed Approach: CAMERA-I	20
3.2	Theory of Electromagnetism for Computer Vision.....	21
3.2.1	Intuitive exercise	22
3.2.2	Electric and Magnetic Monopoles and Dipoles	22
3.3	Potential and Fields Equations Adapted for Computer Vision	24
3.3.1	Geometrical Interpretation of Potentials and Fields.....	25
3.3.2	Convolutions, Potentials and Fields	26
3.4	Application of EM Convolutions	30
3.4.1	Detecting Shape Characteristics.....	30
3.4.2	Magnetic Repulsion for Partial contour Interaction.....	35
3.4.3	Summary of The Advantages of Electromagnetic Convolution Kernels	39
3.4.4	Comparison with the literature	42
3.5	Conclusion.....	43

CHAPTER 4 ARTICLE 2: COMPUTING THE SPATIAL PROBABILITY OF INCLUSION INSIDE PARTIAL CONTOURS FOR COMPUTER VISION APPLICATIONS..... 45

Definitions and acronyms.....	46
4.1 Introduction	46
4.2 Computing the inclusion probabilities with circular paths.....	49
4.2.1 The importance of subsets regions	49
4.2.2 Circular paths between 2 points	52
4.2.3 Intersecting circular arcs	53
4.2.4 Characteristics of the probabilities	55
4.3 Computing the probabilities in an image using EM.....	58
4.3.1 Circular paths transform using EM potential	58
4.3.2 Scalar probability superposition.....	62
4.4 Important properties	67
4.4.1 About the probabilities	67
4.4.2 Additional features	69
4.4.3 3D information estimation from 1D partial contours.....	71
4.5 Conclusion.....	72

CHAPTER 5 ARTICLE 3: FAST AND OPTIMAL LAPLACIAN SOLVER FOR GRADIENT-DOMAIN IMAGE EDITING USING GREEN FUNCTION CONVOLUTION .. 74

5.1 Introduction	75
5.2 Computing the image from its gradient or Laplacian	77
5.2.1 Green's function to solve the Laplacian.....	77
5.2.2 Proof of optimal result for any perturbations in the gradient.....	79
5.2.3 Numerical implementation.....	81
5.3 Applications in computer vision	85

5.3.1	Solving the image Laplacian	85
5.3.2	Gradient-domain image editing	91
5.3.3	Gradient thresholding	93
5.3.4	Gradient domain merging (GDM)	94
5.4	Future work	99
5.5	Conclusion	100
CHAPTER 6 ARTICLE 4: SALIENCY ENHANCEMENT USING GRADIENT DOMAIN EDGES MERGING		101
6.1	Introduction	101
6.2	Saliency enhancement using edges	103
6.2.1	The complete SEE method	103
6.2.2	Salient edge detection	104
6.2.3	Gradient-domain merging	106
6.2.4	The SEE method	109
6.2.5	Evaluation datasets and metrics	115
6.3	Results	116
6.3.1	Results on different images	117
6.3.2	Computation time	118
6.4	Literature comparison and discussion	118
6.4.1	Improvement of the saliency maps	118
6.4.2	Comparison of saliency improvement methods	121
6.5	Conclusion	122
CHAPTER 7 ARTICLE 5: DEEP GREEN FUNCTION CONVOLUTION FOR IMPROVING SALIENCY IN CONVOLUTIONAL NEURAL NETWORKS		124
7.1	Introduction	125

7.2	Methodology	127
7.2.1	Gradient integration and sum (GIS)	127
7.2.2	Implementing the models with the GIS layer.....	130
7.3	Evaluation datasets and metrics	136
7.4	Results	138
7.4.1	Saliency maps.....	138
7.4.2	Validation curves.....	139
7.5	Literature comparison and discussion	143
7.5.1	Training the DSS, DSS-GIS	143
7.5.2	Testing improvement of the GIS layer	143
7.5.3	Literature benchmarking	145
7.5.4	Resistance to noise and low-light.....	147
7.5.5	Computation time	149
7.5.6	Future improvements.....	150
7.6	Conclusion.....	151
CHAPTER 8	ADDITIONAL WORK.....	152
8.1	Prototype and early results for the classification CNN	152
8.2	Generative networks.....	156
CHAPTER 9	GENERAL DISCUSSION AND FUTURE WORK	157
9.1	Achieving the research objectives	157
9.2	New computation tools for machine vision.....	160
9.3	Improving deep networks.....	161
9.3.1	Improvement of the saliency network.....	161
9.3.2	Enabling unlimited receptive field	162

9.4	Industrial applications and patents	163
9.5	Future work	164
9.6	Limitations	165
9.6.1	Mathematical limitations.....	165
9.6.2	Practical limitations.....	167
9.7	Thesis outcomes	168
9.7.1	Tools.....	168
9.7.2	Deliverables.....	170
9.7.3	Scientific outcomes	171
CHAPTER 10 CONCLUSION AND RECOMMENDATIONS.....		173
BIBLIOGRAPHY		175
APPENDICES.....		187

LIST OF TABLES

Table 3.1 : Potential and Field Characteristics at Different Regions of a Shape Filled with Monopoles, for $n > 2$	26
Table 3.2 : Percentile Thresholds Used for the Discovery of the Regions of Interest	33
Table 3.3 : Qualitative Comparison Between Different Image Analysis Methods.....	42
Table 4.1 : List of fundamental properties for the consistency of the probabilities.....	58
Table 7.1 : Side layer information of HED and DSS architectures given by $n, k \times k$, where n is the number of output channels and $k \times k$ is the size of the kernel. “Layer” is the name of the layer from the VGGnet-16 whose output is connected to a side layer. “1”, “2” and “3” represent the 3 layers for each side output. “1” and “2” are followed by a ReLU operation. If a GIS layer is added, $no = 3$, otherwise $no = 1$	134
Table 7.2 : Comparison between the percentage testing results of HED and our HED-GIS. The * means that a denseCRF layer is added at the output. The best value in each category is highlighted in bold (ignoring GIS*). The values are in percentages.	144
Table 7.3 : Comparison between the testing results of DSS and our DSS-GIS. The * means that a denseCRF layer is added at the output. The best value in each category is highlighted in bold (ignoring GIS*). The values are in percentages.	145
Table 7.4 : Comparison of the proposed DSS-GIS and HED-GIS approaches (grey rows) with other saliency algorithms proposed in the literature. The best result of each column is highlighted in bold. The values are in percentages.	146
Table 7.5 : Comparison of the DSS and HED approaches with and without the proposed GIS layer when a 30% salt-and-pepper noise is added to the test set. The best result of each column is highlighted in bold. The values are in percentages.	148
Table 7.6 : Comparison of the DSS and HED approaches with and without the proposed GIS layer when a the brightness is reduced by 80% to simulate low-light pictures. The best result of each column is highlighted in bold. The values are in percentages.	149
Table 9.1 : Tools developed during the thesis in different languages and libraries	169
Table 10.1 : Qualitative Comparison Between the Partial contours Presented in Figure 10-6....	202

LIST OF FIGURES

Figure 1-1 : Overview of the objective and methodology of the thesis	3
Figure 2-1 : Example of saliency for a person image from the ECSSD dataset [68]; (a) Original image; (b) Ground-truth saliency map; (c) Saliency map produced by the DRFI method [67]; (d) Saliency map produced by the DSS method [26]......	11
Figure 2-2 : Image and channel sizes throughout the VGGnet-16 given by width \times height \times channels [10,83]	14
Figure 2-3 : Convolution kernel sizes given by width \times height, number of neurons [10,83]	15
Figure 2-4 : Illustration of the InceptionNet v1 architecture, known as GoogLeNet [20,85]......	15
Figure 2-5 : Architecture of the ResNet neural network [84,85]	16
Figure 3-1 : Static electric potential and field of a: (a) positive monopole. (b) negative monopole	22
Figure 3-2 : Electric Potential and field for static monopoles placed as (a) a dipole. (b) a small chain of simple dipoles. (c) a horizontal and a vertical dipole, equivalent as 2 dipoles at 45° . (d) a long chain of simple dipoles. (e) simple dipoles in parallel.....	23
Figure 3-3 : Potential and field with $n = 3$ for positive monopoles placed on (a) A circle. (b) A corner.....	25
Figure 3-4 : Example of convolution kernel for a particle potential matrix Pe of size 7×7 ; (a) Euclidian distance from center r . (b) Potential of a centered monopole $Pe = Ve$, $n = 3$	27
Figure 3-5 : Steps to calculate the normalized potential kernel for a dipole (a) Positive and negative monopoles at 1 pixel distance. (b) Potential kernel Pe . (c) Dipole potential kernel $Pdipx$ resulting from the convolution of image “a” with kernel “b”.	28
Figure 3-6 : Calculation of the potential and field of an image (a) Monopoles in the image. (b) Potential kernel Pe . (c) Total potential Ve . (d) Horizontal field E_{ex} . (e) Vertical field E_{ey} . (f) Field norm Ee and direction.....	29

- Figure 3-7 : Steps to calculate the magnetic PF of an image (a) Dipoles in the image. (b) Horizontal dipole potential kernel Pmx . (c) Total potential Vm . (d) Horizontal field Emx . (e) Vertical field Emy . (f) Field norm Em and direction.....30
- Figure 3-8 : (a) Special shape with the white region being a uniform density of charge, used to compute the following PF with $n = 3$. (b) The potential Ve . (c) The field Ee . (d) The potential squared only on the contour $VeonC2$. (e) The field squared only on the contour $EeonC2$...32
- Figure 3-9 : RoI found on a complex shape using a contour analysis by potential and field thresholds. (a) Concave regions. (b) Convex regions. (c) Flat regions. (d) Regions near the CM. (e) Regions far from the CM. (f) Regions inside the shape.33
- Figure 3-10 : RoI found on a complex shape (filtered with a twirl, twist and wave distortion) using a contour analysis by potential and field thresholds. (a) Concave regions. (b) Convex regions. (c) Flat regions. (d) Regions near the CM. (e) Regions far from the CM. (f) Regions inside the shape.....34
- Figure 3-11 : EM potential V and field E generated by a 3D mug, with different values of n . (a) $VonC2$ with $n = 3$. (b) $EonC2$ with $n = 3$. (c) $VonC2$ with $n = 4$. (d) $EonC2$ with $n = 4$35
- Figure 3-12 : Potential Vm resulting of the convolution of a dipole perpendicular to the partial contour lines. (a) Partial contour with low resolution 64x64. (b) Dipole with $n = 3$ and low resolution. (c) Dipole with $n = 2$ and low resolution. (d) Partial contour with high resolution 512x512. (e) Dipole with $n = 3$ and high resolution. (f) Dipole with $n = 2$ and high resolution.....36
- Figure 3-13 : Potential Vm of a circular partial contour magnetized perpendicular to their orientations. (a) Circle arc of 90° , with $n = 2$. (b) Circle arc of 270° , with $n = 2$. (c) Circle arc of 360° , with $n = 2$. (d) Circle arc of 90° , with $n = 3$. (e) Circle arc of 270° , with $n = 3$. (f) Circle arc of 360° , with $n = 3$37
- Figure 3-14 : PF computed from the initial partial contour, with $n = 2$ and the dipole perpendicular to the partial contours. (a) Initial partial contour. (b) Potential of attraction Vm . (c) Potential of repulsion Vm . (d) Field of attraction $Em0.5$. (e) Field of repulsion $Em0.5$38

- Figure 3-15 : Partial contours for the number “2” at the left, with the potentials V_m of dipoles perpendicular to the partial contours, with $n = 2$. (a) Clean partial contour. (b) Deformed partial contour. (c) Heavily distorted partial contour.39
- Figure 3-16 : Contour approximation via Fourier descriptors. (a) Fourier descriptor with 4 harmonics. (b) Fourier descriptors with 32 harmonics. [37,38].....43
- Figure 4-1 : Definitions of different concepts. (a) Image of an elk from BSD500 dataset. (b) Edges computed using the Sobel algorithm. (c) The contour of the elk. (d) Partial contour (stroke) along with 2 possible paths that close the partial contour.47
- Figure 4-2 : Example of a partial contour S between points γ_i, f , closed by a path S_n to generate the region R_n containing the point γ_{in} but excluding γ_{out} . S is an existing partial contour and does not have restrictions. S_n is the generated path used to close S , thus S_n must be non-self-intersecting, convex and smooth.50
- Figure 4-3 : Example of 7 paths S_n between points γ_i and γ_f , with starting angles $\beta_1 \rightarrow 5 +$ and $\beta_1 \rightarrow 2 -$, such that R_n , the region between S_n and S , is a subset of R_{n+1}51
- Figure 4-4 : Example of a circular path between points γ_i and γ_f , with a starting angle β53
- Figure 4-5 : Example of 2 regions $R(\beta_1, 2+)$ formed by the closure of the path S with the circular arcs $SC\beta_1, 2 +$ 53
- Figure 4-6 : Example of (a) a partial contour S that intersect a circular arc SC at the point $\gamma \times$. (b) The region inside the closure of the partial contour S with the sub-paths $SC -$ and $SC +$. ..54
- Figure 4-7 : Example of (a) A complex intersection between S and SC . (b) The regions that do not respect the logical operation are eliminated. (c) The region R is formed with a union of the remaining regions.55
- Figure 4-8 : Complementarity of the enclosure probability across the path S . (a) 2 points at an opposed side of S . (b) Multiple complementary points, at opposed sides of S , but with an infinitesimal distance.....57
- Figure 4-9 : A line on the x axis, composed of dipoles parallel to the y axis.....59

- Figure 4-10 : Example of equipotential lines of V_m (green and pink) computed on 6 different partial contours (dark grey), with the light grey lines being the perfectly circular equipotential lines of equations (40) and (41)62
- Figure 4-11 : Algorithm used for repulsion optimization by flipping the magnetic orientation of each individual or group of sub-partial contours Gk64
- Figure 4-12 : (a) Artificial image composed of different nearby shapes (b) Extracted partial contours using Canny [63]; (c) Resulting V_m in the initial orientation; (d) resulting V_m after the repulsion optimization.64
- Figure 4-13 : (a) Artificial image composed of different adjacent shapes (b) Extracted partial contours using Canny [63], with the double boundaries in green; (c) Resulting V_m without the double boundary; (d) resulting V_m with the double boundary.65
- Figure 4-14 : The algorithm used for the image splitting into multiple sub-images.....66
- Figure 4-15 : Example of image splitting process; (a, b, e) The temporary states of the splitting; (c, d, f, g) The final set of split potentials v_{mi}67
- Figure 4-16 : Comparison of the original images with the probabilistic reconstruction. (a, c, e) Original synthetic image composed of different shapes with the partial contours (orange); (b, d, f) Probabilistic weighted reconstruction based on the partial contours (orange).72
- Figure 5-1 : Process summary of the gradient domain image editing.....86
- Figure 5-2 : Computation time (ms) in logarithmic scale for a single channel gradient solving of resolution of 801x1200, including the preparation time. The Perez [57] and Tanaka [61] methods have no GPU implementation. The Pytorch* implementation is tested on batches of 100 images, and the total time is divided by 100.90
- Figure 5-3 : Comparison of the RMSE between E_c and E_p , where the thresholds are the perturbation used to generate E_p . The displayed values are the mean of the RMSE on the 1000 images of the ECSSD dataset [68].91
- Figure 5-4 : Example of Poisson Blending application; (a) Stamp to copy, with the red-dotted lines being the cropping region and the blue dotted circle being a region of the stamp that is

accidentally cropped; (b) destination image; (c) Poisson blending from Perez algorithm [57,118]; (d) Proposed GFC blending.....	92
Figure 5-5 : Example of gradient thresholding and solver steps. (a) Original image; (b) Gradient \mathbf{E} ; (c) Thresholded gradient at 10%; (d) Solved image $I_c, corr$	93
Figure 5-6 : Examples of solved images $I_c, corr$ after the gradient threshold at 10%. (a) Image of a castle; (b) Solved castle image after 10% gradient threshold; (c) Image of a leopard; (d) Solved leopard image after 10% gradient threshold.	94
Figure 5-7 : Examples of the steps involving the GDM equation (73). (a) Original image; (b) \mathbf{E} : Gradient; (c) $\mathbf{E}p$: Gradient merged with the SE method [64] and $\alpha = 0.5$; (d) Solved image.	95
Figure 5-8 : Example of GDM using equation (73) with a random forest edge detector [64] and varying parameter α	96
Figure 5-9 : Examples of solved images $I_c, corr$ with a perturbed gradient from equation (73) with edges information from SE method [64] and RCF method [25] and $\alpha = 0.5$	97
Figure 5-10 : Comparison of edge merging methods. (a) Original image; (b) Saliency sharpening from Bhat et al. [58]; (c) Enhancement using our GDM method with $\alpha = 0.5$ and SE edge detector.	97
Figure 5-11 : Examples of solved images $I_c, corr$ with a perturbed gradient from equation (73) with edges information from SE method [64] with NMS and $\alpha = 0.5$	99
Figure 6-1 : Diagram of the complete SEE method, which allows improving the saliency via a combined image preprocessing and saliency map postprocessing. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example where SI is the original saliency and $SIORRC$ is the same saliency improved using our SEE method.	104
Figure 6-2 : Comparison of SE and RCF edge detection methods trained or finetuned on the BSDS500 or MSRA10K datasets, with the number of iterations of retraining, using 2 example images from the ECSSD dataset. The salient edges will be used as inputs for the proposed SEE method.....	105

Figure 6-3 : Diagram of the core GDM function, allowing to merge image/saliency information with edges information by passing to the gradient domain and solving the Laplacian equation to go back to the image domain.	109
Figure 6-4 : Diagram of the SEE-Post method, which allows to post-process a saliency map based on the salient edge detection. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example.	110
Figure 6-5 : Diagram of the SEE-Pre method, which allows preprocessing an image based on the salient edge detection for a more accurate saliency map. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example.	113
Figure 6-6 : Smooth-step function (85) with different values of K	115
Figure 6-7 : Comparison of the results from 5 different SoA methods (MC, RBD, DRFI, DCL, and DSS) and their improvement using our SEE algorithm highlighted in green. The examples are chosen as some of the most difficult images in the datasets.	117
Figure 6-8 : Comparison between the regular saliency methods, the enhanced saliency using our SEE-Post algorithm and the enhanced saliency using our SEE algorithm. A higher percentage is better for AUC , Fm , PR and $Pmax$, but a lower percentage is better for MAE . The 3 datasets used are ECSSD (E), PASCAL-S (P-S) and DUT-OMRON (D-O).....	120
Figure 6-9 : Precision-recall curves comparing 6 different methods (DSR, RBD, DRFI, MC, DCL, and DSS) with and without our SEE method on 3 different datasets.	121
Figure 6-10 : Comparison of the improvement over Fm and MAE for different SoA saliency improvement methods.	122
Figure 7-1 : Graph summary of the gradient integration and sum (GIS) layer, which outputs n channels from $3n$ inputs.....	130
Figure 7-2 : The HED [66] and DSS [11,26] architectures nested upon the pre-trained VGGnet-16 [10] network, with a total of 6 side layers. The red arrows are the short connections implemented by the DSS model [11,26]. Our contribution is the GIS at the end of each side-layer, which requires the layer sideX_3 to output 3 channels instead of 1.	132

- Figure 7-3 : Closer view on the integration of the GIS layer inside the DSS architecture. For the HED architecture, the GIS layer is placed directly after the channel split since there are no short connections..... 133
- Figure 7-4 : Example of the inputs of the GIS layer coming from a fully trained HED-GIS network. S is expected to be in the saliency domain; Ex, Ey are expected to represent the 2 components of the gradient domain..... 135
- Figure 7-5 : Test results comparison between the DSS and HED methods with and without the proposed GIS layer..... 141
- Figure 7-6 : Comparison of the validation curves of different models for the optimally found parameters. The validation performance is computed every 500 iterations on the full validation set. 142
- Figure 7-7 : Comparison of 6 validation curves in orange of our implementation of the DSS model, and 6 validation curves in blue of our DSS-GIS model. The curves are smoothed using exponential smoothing with a factor 0.9, and the x-axis represents the number of iterations. 142
- Figure 7-8 : Precision-Recall curves (top row) and true-positive/false-positive curves (bottom row) for the 3 test datasets 146
- Figure 7-9 : Test results comparison between the DSS and HED methods with and without the proposed GIS layer, when the test set is modified with noise or reduced brightness. 148
- Figure 7-10 : Performance impact of adding noise to the testing set for the DSS and the proposed DSS-GIS methods. 149
- Figure 8-1 : Inception module of the Google-net [20] with an added GDI2 or GDI3 layer. The Conv + R indicate a convolutional layer with a Relu activation function, the $n \times n$ means that the kernel size is n and the mS means that the stride is m 153
- Figure 8-2 : Internal structure of the GDI2 which splits the data into 2 inputs for the GFC; and GDI3 which splits the data into 3 inputs for the GDM or GIS layers. The dx and dy represent numerical derivatives and the Conv-layer does not use an activation function or bias; the $n \times n$ means that the kernel size is n and the mS means that the stride is m 154

Figure 8-3 : Validation accuracy on the MNIST dataset for a smaller Google-net with 2 inception layers. The blue line is the standard network and the orange line is the same network with the added GID2 in each inception layer.	155
Figure 9-1 : Example of saliency maps produced with a single layer of convolution network which is able to perfectly detect the salient edges, but not the regions within the edges. The images are not real results, they are produced by an image editing software. (a1) Original image; (a2) True saliency value; (b1) Detected normalized saliency features with the small receptive field; (b2) Final saliency after thresholding; (c1) Detected normalized directional gradient of saliency features with the small receptive field; (c2) extended features using the GFC; (c3) Final saliency after GFC and thresholding.	163
Figure 9-2 : Example of documentation used for our <i>green_function_torch</i> module.	170
Figure 9-3 Our contribution, in terms of categories of possible operations inside CNN.	172
Figure 10-1 : Neural network architecture with the neurons being the yellow circles, with a view inside the neurons.	190
Figure 10-2 : Static electric potential and field of a: (a) positive monopole. (b) negative monopole	192
Figure 10-3 : Electric Potential and field for static monopoles placed as (a) A simple dipole. (b) A small chain of simple dipoles. (c) A horizontal and a vertical dipole, equivalent as 2 dipoles at 45° . (d) A long chain of simple dipoles in series. (e) A long chain of simple dipoles in parallel.	193
Figure 10-4 : Potential and field for circles, with $n = 3$, for (a) positive monopoles. (b) parallel dipoles. The green part is the positive charges/potential, and the purple part is the negative charges/potential.	196
Figure 10-5 : Potential and field for corners of (a) positive monopoles. (b) parallel dipoles.	197
Figure 10-6 : Example of different symmetric paths between points γ_i and γ_f , with a starting angle β	201
Figure 10-7 : Example of a circular path between points γ_i and γ_f , with a starting angle β	204
Figure 10-8 : Demonstration that the potential V_m has circular equipotentials	206

LIST OF SYMBOLS AND ABBREVIATIONS

The list of symbols and abbreviations presents the symbols and abbreviations used in the thesis or dissertation in alphabetical order, along with their meanings. First, we present the list of symbols, then the list of abbreviations.

Latin symbols

C, C_0	Edge intensity computed from an edge detection method
dip	Dipole
e, m	Electric (e) or Magnetic (m)
$\mathbf{E}_{e,m}$	Virtual Vector field associated to electric e or magnetic m charge
\mathbf{E}	Electric field, being the gradient of V_E or V_m
\mathbf{E}_c	Reconstructed conservative electric field
\mathbf{E}_p	Perturbed non-conservative electric field
F	Density correction factor of I
F_m	F-measure on a given dataset
G	Binary ground truth of the saliency
G_k	A possible group of different s_i or S
I	Image matrix, with values between -1 and +1
I_0	Input image normalized in the range $[0, 1]$
I_R	The resulting image after solving the Laplacian L_p
$K_{I \rightarrow E}$	Complex kernel used to compute the gradient \mathbf{E} from an image I_0
$K_{E \rightarrow L}$	Complex kernel used to compute the Laplacian L_p from a gradient \mathbf{E}
K_{∇^2}	Convolution kernel for the Laplacian operator
\tilde{K}_{∇^2}	Zero-padded Laplacian kernel
K_{dx}	Derivative kernel in the x direction

K_θ	Complex derivative kernel in 2D
L_p	Laplacian of the image I_0
M	Binary mask generated by a threshold on S
M_E	Function used to merge the norm of the gradient with the edges
M_θ	Function used to merge the orientation of the gradient with the edges
N	Number of partial contours, or Total number of pixels in an image
n	Number of spatial dimensions for the virtual potential ($n = 2$ for an image)
onC	Only values on the Contour
P	Precision of S against G
P_{\max}	Maximum value of P
$P_{e,m,dip}$	Virtual Potential kernel of a monopole or dipole
P_e	Electric potential by a single monopole
P_{dip}^θ	Complex dipole potential
P_S	Probability of being enclosed in S
P_S^\pm	The value of P_S associated to V_m^\pm
\overline{PR}	Average of the precision-recall curve
$P_{I_0 \rightarrow 1}$	Preparation step for the image
$P_{C_0 \rightarrow 1}$	Preparation step for the edges
$q_{e,m}$	Virtual Charge
r	Euclidean distance
R	The region inside a closed S , or Recall of S against G
S	Partial contour, defined as a partial contour, or Saliency map defined in the range
$[0, 1]$	
S_C	Circular partial contour, or Saliency map with enhanced contrast

S_C^\pm	The sections of S_C defined with β^\pm
S_n	The n-sphere area constant ($S_2 = 2\pi$)
s_i	A part of the partial contour S
t	Time for parametric functions
$t_{i,f}$	Initial and final time
U	Any possible potential, meaning that ΔU is any possible field
$V_{e,m}$	Virtual Potential associated to electric e or magnetic m charge
V_m^\pm	Region where the V_m is expected to be positive/negative
V_E	Potential, often represents the signal or computed image
$V_{E,corr}$	Potential corrected to preserve colours and contrast
V_{mono}	Green's function, equivalent to electric potential by a single monopole
$V_{mono}^{\mathcal{F}}$	Optimal Green's function in the Fourier domain
\mathbf{V}_{dip}	Gradient of V_{mono}
V_{dip}^θ	Complex dipole potential
W_S	Weighted probability
w_S	Weighted probability of a sub-image
x_m	Cartesian coordinate in the dimension m
x, y	Horizontal and vertical axis

Greek or scripts symbols

α	Factor between 0 and 1 to preserve the edge invariance
β	The starting angle of a symmetric S , or parameter of the F-measure set to 0.3
β^\pm	The starting angle from one side of S , or the other side
δ	Numerical Dirac's delta

δ	Zero-padded numerical Dirac's delta
θ	Orientation matrix perpendicular to the partial contours in I
θ_C	Orientation of the edges C
θ_E	Orientation of the gradient E
γ	A specific point in space
$\gamma_{i,f}$	The starting and ending points of S
$\mathcal{E}_{e,m}$	Electromagnetic vector field according to physics $[\text{V m}^{-1}]_e, [\text{V s m}^{-2}]_m$
$\mathcal{V}_{e,m}$	Electromagnetic potential according to physics $[\text{V}]_e, [\text{V s m}^{-1}]_m$
\mathcal{F}	Fourier transform
\mathcal{F}^{-1}	Inverse Fourier transform
$\Re(), \Im()$	Real part \mathcal{R} or Imaginary part \mathcal{I} of any complex value
σ	Standard deviation
$\Omega(f)$	Variance of f

Operators

$\Delta^{x,y}$	Numerical derivative kernel in \hat{x} or \hat{y} direction
∇	Gradient operator
∇^2	Laplacian operator
$\nabla \cdot$	Divergence operator
\cdot	Scalar product
$*$	Convolution operator
\circ	Hadamard product (Element-wise multiplication)
\subset	Subset operator, used to indicate γ is inside R
\cap	Intersection (AND operator)

\cup	Union (OR operator)
$\cup_i a_i$	Union of all the elements a_i
\pm	Either sum or subtraction, to be decided
$-$	Mean value
$!$	Logical NOT operator

Abbreviations

AUC	Area under the true-false-positive curve
CAMERA-I	Convolutional approach of magnetic and electric repulsion to analyse an image
CNN	Convolutional neural network
CPU	Central processing unit
CV	Computer vision
DSS-GIS	Deeply supervised saliency with gradient integration and sum
EM	Electromagnetic OR Electromagnetism
FFT	Fast Fourier transform
GDIE	Gradient domain image editing
GDM	Gradient-domain merging
GIS	Gradient integration and sum
GF	Green's function
GFC	Green function convolution
GPU	Graphic processing unit
MAE	Mean absolute error between
NN	Neural network
PIIPE	Probability of inclusion inside partial edges
Post	For postprocessing

Pre	For preprocessing
RGB	Red green blue
RMSE	Root-mean-squared-error
SEE	Saliency enhancement using edges
SoA	State of the art
SOD	Salient object detection

LIST OF APPENDICES

APPENDIX A	BENCHMARKING PARAMETERS.....	187
APPENDIX B	STANDARD NEURAL NETWORKS.....	189
APPENDIX C	GENERAL ELECTROMAGNETISM.....	191
C.1	Supplementary Nomenclature.....	191
C.2	Monopoles and Dipoles.....	191
C.2.1	Electric monopoles.....	191
C.2.2	Electric dipoles.....	192
C.2.3	Magnetic charges and dipoles.....	193
C.3	Mathematical Laws of EM.....	194
C.3.1	Maxwell's Equations.....	194
C.3.2	Adaptation of Maxwell's equations for Computer Vision.....	195
C.4	Geometrical Interpretation of Maxwell's Equations.....	196
C.4.1	Closed shapes with particles on the contour.....	196
C.4.2	Corners with particles on the contour.....	197
C.5	Partial contour Manipulations.....	197
C.5.1	Grow and unite contour regions.....	198
C.5.2	Finding Partial contour Orientation.....	199
APPENDIX D	GENERAL ELECTROMAGNETISM.....	201
D.1	Supplementary nomenclature.....	201
D.2	Paths characteristics.....	201
D.2.1	Characteristics of the paths between 2 points.....	201
D.2.2	Choosing the circle, rejecting the parabola.....	202
D.2.3	Circular path parameters.....	203

D.3	Electromagnetic potential.....	204
D.3.1	Elliptical potentials and paths	204
D.3.2	Convolution alternative	205
D.3.3	Demonstration that equipotential lines are circular.....	205

CHAPTER 1 INTRODUCTION

Since the year 2000, there has been an exponential rise in the sale of consumer digital cameras [1] followed by the bigger rise of smartphones equipped with digital cameras [2] and the exponential increase in the commercial robotics market [3]. Those factors led directly to an explosion in the computer vision (CV) and artificial intelligence (AI) market, both in robotics and in data analysis [4]. For example, our research lab uses CV to perform airplane topological optimization [5], intelligent robot control for the disabled using eye-tracking [6,7] and automated drone control.

Then, the years 2010s have been revolutionary in terms of image understanding thanks to the rise of machine learning algorithms and convolutional neural networks (CNN). The CNN was initially able to outperform any standard algorithm for image classification purposes [8–10]. Then, they were used to solve the binary problems of CV, such as edge detection, skeleton extraction and saliency [11], and are now performing at near human level.

Chapter 1 In the field of machine learning and CV, many successful methods were based on the biological observation of nature. For example, the structure of the CNN is heavily inspired by the way the frontal cortex analyses images [9,12]. However, there are limited uses of physically inspired models. For example, the usage of electromagnetic (EM) based fields is limited to quadrupole text orientation [13] and the gravity-based edge detection [14]. Although those methods were inspired by a physical model and they initially performed well, they were soon outpaced by competing methods. One of the reasons is that they used the potentials and fields convolutions in an intuitive way such as blurring filters and derivative filters. The current thesis differs since it thoroughly studies the mathematical behavior of EM and GF for image convolutions.

1.1 Objectives

The main objective of the current thesis is to **develop electromagnetic (EM) convolutions and Green's functions (GF) convolutions to be used in Computer Vision and convolutional neural networks (CNN).**

The main objective can be divided into the following sub-objectives (Obj):

- Obj - 1. Develop a mathematical and intuitive understanding of the behavior of EM and GF convolutions in an image.**

This objective allows to better guide the research decisions and to better understand the results. It was fundamental for the next sub-objectives.

Obj - 2. Use the GF convolutions to reduce the computation time and numerical error of the EM and allow fast and efficient gradient-domain image editing.

This objective allows editing an image in a non-intuitive way by enhancing or reducing the gradient of the image at specified locations.

Obj - 3. Use the GF to improve the results of CNN for salient object detection and digit classification.

This objective allows improving the extracted features by using the properties of conservation of energy and smoothness of EM. It also allows the current work to have a broader impact on the CV community since an important part of the research focuses of CNN.

1.2 Methodology

To answer the objectives given previously, a total of 5 papers were written and are included as chapters in the current thesis. This section will present a summary of the methodology and the articles developed to answer each of the sub-objectives. First, an overview of the objectives of the thesis is presented in Figure 1-1, followed by the detailed methodology below.

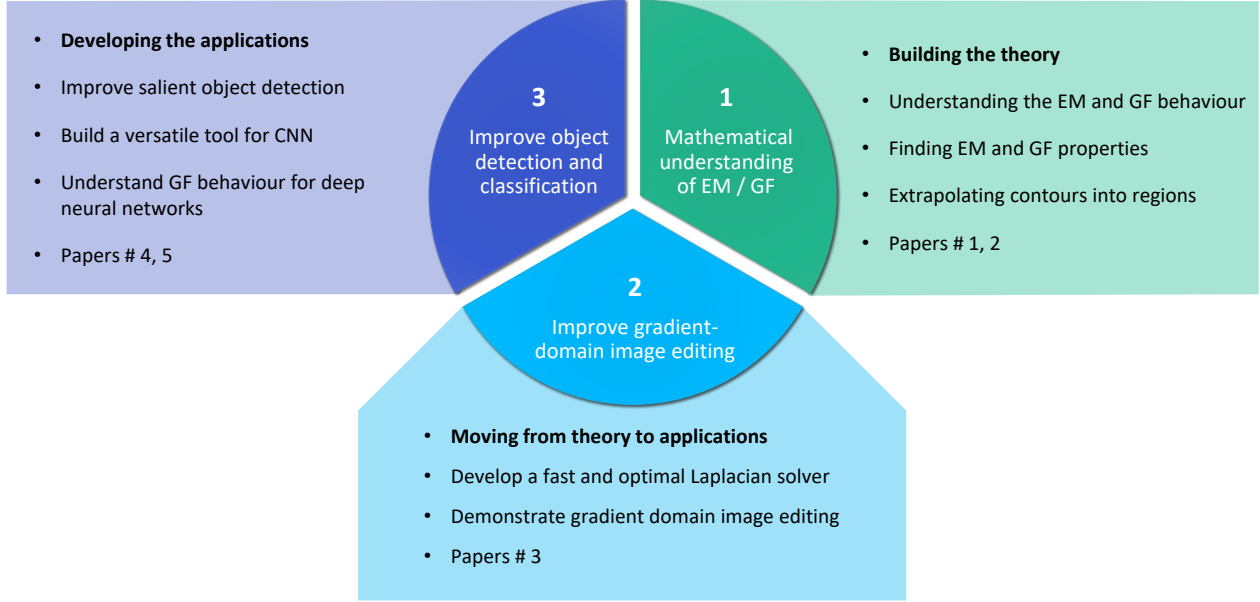


Figure 1-1 : Overview of the objective and methodology of the thesis

Obj - 1. The thesis first studies how Maxell's equations of EM behave in an image using their numerous physical, geometrical and mathematical properties. The study is presented in our first paper [15] in Chapter 3. It computes the EM potential and field using numerical convolutions and performs a qualitative analysis of the results. Then, the second paper [16] presented in Chapter 4 demonstrates mathematically that dipole potentials can be used to compute the probability of belonging inside partial contours. For example, the paper showed how the resulting potential allows reconstructing the shapes when only incomplete contours are provided. These two papers were fundamental for answering the first objective, specifically to develop a mathematical and intuitive understanding of the behavior of EM convolutions in an image.

Obj - 2. After demonstrating the usefulness of EM potential and fields, the GF is developed in the paper [17] presented in Chapter 5. The GF reduces the numerical error to almost zero and improves the computation time by a factor 4 compared to EM. In fact, we have proven mathematically that, when a non-conservative perturbation is added to a gradient, the GF convolution (GFC) is the least-error gradient or Laplacian solver. Plus, it is shown to perform well on multiple tasks of gradient domain image editing. Hence, this paper answers the second objective of the current thesis, specifically to reduce the computation time and numerical error of the EM potentials while developing a gradient domain editing method.

Obj - 3. Using the efficient GFC Laplacian solver, the current objective is to improve salient object detection and neural network using the gradient domain. At first, the objective focuses on salient object detection, with the paper [18] presented in Chapter 6. In this work, we developed a method that combines the output of a saliency network with the output of an edge detection network using a novel method we called gradient domain merging (GDM). The GDM uses the gradient domain to merge features of different nature (edges vs regions), then uses a GFC to solve the perturbed gradient. It showed to be fast and effective at improving the saliency maps of different methods. Then, the next paper [19] presented in Chapter 7 shows that another GFC-based operation, called gradient integration and sum, can be added inside different CNN to improve the testing saliency maps further, to reduce the training convergence time, to reduce the model overfitting and to increase the robustness against parameter initialization and noise. Finally, additional prototypes are developed in section 8.1 to demonstrate that GFC can be added within the Google-net [20] to improve its training time and testing performance on the MNIST dataset [8]. These 3 chapters and section allowed to answer the third objective of the current thesis, specifically to improve the results of CNN on image analysis using GF convolutions (GFC).

1.3 Overview of the thesis

In the CV field, the current thesis positions itself mostly in the fields of mathematical imaging, feature detection via machine learning, and image processing. For the mathematical imaging, the current thesis developed EM and GF convolutions for image editing, demonstrated that they can be used to compute contour inclusion probabilities [16] and that they are the least-error gradient and Laplacian solver. For the feature detection, the presented work explored many different areas such as shape analysis [15,21], saliency detection improvement [18] and deep learning classification. In general, our approach was fast to compute and able to improve many state-of-the-art methods. For image processing, the proposed GFC showed to be faster and have less error than competing methods of gradient domain image editing [17].

The work done in the current thesis allowed our team to submit 3 patents with support from Polytechnique Montreal and the technology transfer company *Univalor*. The first patent “Object analysis in images using electric potentials and electric fields” [21] is already accepted, but the

second and third have been submitted at the time of writing and are pending approval from the patent office.

Chapter 2 will present a detailed literature review of the different methods related to the proposed approach.

Chapters 3-7 will present each of 5 different scientific papers that are submitted to scientific journals, conferences or archives.

Chapter 3 is about the general understanding of CAMERA-I since it presents Maxwell's equations, it finds the potential and field equations for an n -dimension system, and it focuses on the 2D properties of monopoles and dipoles [15].

Chapter 4 focuses on the mathematical demonstration of how EM potentials and fields allow to numerically compute the probability of inclusion inside partial edges (PIIPE), how to use it on images composed of edges and partial contours and how do these edges and partial contours interact [16].

Chapter 5 follows by demonstrating that the EM kernels can be used to perform a Green's function convolution (GFC), thus solving the Laplacian or Gradient in a fast and robust way which allows for fast gradient-domain image editing (GDIE) [17].

Chapter 6 shows that GDIE can be used for saliency enhancement using edges (SEE), which allows using gradient domain merging (GDM) between the 1D edge information with the 2D saliency information. Hence, it was possible to improve the results of any high-performance saliency algorithm [18].

Chapter 7 demonstrates that the proposed gradient and integration and sum (GIS) layer enhances standard deep saliency models since it improves the repeatability of the training, reduces the overfit, enhances the testing results and improves the stability to noise.

Finally, Chapter 9 includes a critical review of the thesis results and a discussion of its limitations. Then, the chapter proposes future improvements and possibilities of research, including how to use the GFC in different types of 2D CNN.

CHAPTER 2 LITERATURE REVIEW

The scope of the current thesis covers different fields of computer vision (CV), most notably nature-based algorithms, shape and partial contour analysis, feature detection, gradient-domain image editing, and convolutional neural networks (CNN). Hence, this section will present a detailed literature review for each of the covered CV fields.

2.1 Nature-based algorithms

Since the electromagnetism (EM) approach proposed research project is based on physics principle, this section will cover some other nature-inspired techniques in CV.

2.1.1 Biology inspired techniques

For the biology-inspired techniques, the most prominent one is the neural networks (NN), more specifically the convolution neural networks (CNN) [22–24], which mimics how the human frontal cortex works. Currently, the CNN are the most performant methods in classification [8,22,24], edge detection [11,25] and salient object detection [11,26,27]. More details are given in the section “2.5 The convolutional neural network”.

There are other biology inspired techniques, like those based on population evolution, but they are principally used for parameter optimization in CV [24], which is not relevant in the case of this research project.

2.1.2 Physics-inspired techniques

The use of physics inspired techniques in CV is not as present in literature as biology, due to the success of NN and NN-inspired approaches. Some examples of physics inspired techniques include the entropy used to analyze the information of an image [24], watershed droplets for edge detection [28] and force vector fields used for active contours [29].

The main methods related to my Ph.D. research project are the quadrupole convolution, used to define the orientation of contours [13] and the edge detection using gravitational fields (which is mathematically similar to electric fields) [14]. Although those 2 works proved to be efficient new ways of analyzing images, they did not see the full possibilities of using EM fields, nor did they use any EM potential or Green’s function (GF). This research thesis aims to go a lot further in

exploring a multitude of possibilities using the laws of EM as described by J.C. Maxwell [30–32] and the properties of GF [17,33] for shape/partial contour analysis, gradient/Laplacian solver and saliency improvement.

2.2 Shape and partial contour analysis

From the early days of computer vision, one of the early tasks was to analyze shape profiles and partial contours using different tools. Those tools include morphological operations [34,35], boundary and corner detection [36], skeleton extraction [36], elliptic Fourier transforms [37,38] and fractal dimensions [39]. However, most of those tools are considered early vision and are minimally used in most recent papers and advanced technique [23,34], especially since the rise of deep learning. This section will show that the proposed approach, based on electromagnetism, distinguishes itself from the rest of the literature by providing a fast, robust and unique way of analyzing a shape or partial contour. In fact, the proposed approach still performs well alongside deep learning algorithms.

2.2.1 Electromagnetic convolutions

For a great part of the shape and partial contour analysis, convolution kernels were always among the favored method of information extraction, with multiple usages in noise removal [40], defect detection [41,42], image segmentation [29,43], edge detection [14], machine learning [24,44,45], etc. They are known to be easy to implement, fast to compute, are translation agnostic and are available in most computer vision libraries such as MATLAB® (Mathworks, USA) [36] and OpenCV [35]. To extract the features of the shapes, each convolution uses a kernel that “slides” over an image to apply local arithmetic’s on neighbor pixels [23]. Usually, the convolution kernels are small since they only require the information of nearby pixels, with usual maximum sizes of 5×5 [20], 7×7 [14], 9×9 [41], 11×11 [45]. However, some cases such as the EM or GF kernels developed in our work [15,16,21] or other physics-based methods [29,43], the kernels are bigger than the image.

With such a big kernel size, fast Fourier transforms (FFT) are used to speed up the computation of the convolution [23]. For an image of n pixels and a kernel of m element, a standard convolution has a time complexity of $O(n \cdot m)$ but a FFT based algorithm has a time complexity of $O(n \log(n) + m \log(m))$ [23]. Hence, if we suppose that the kernel is the same size as the image

($m = n$), then the time complexity is $O(n^2)$ for a standard convolution and $O(n \log(n))$ for the FFT.

The proposed image analysis with CAMERA-I and GF goes further than the methods reported in the literature by providing multiple symmetrical or asymmetrical kernels that are scale invariant and rotation invariant [15,21]. Hence, they give the same result, no matter the resolution of the image. This seems to contradict the scale-space theory of computer vision which states that different information is available at different scales [46]. However, many applications require scale invariance such as contour completion [47] and wavelet analysis [48]. Plus, the rotation invariance is important since many applications use the same filters in different orientations to detect rotation invariant features for defect detection [41] or for edge detection [49]. Furthermore, some advanced machine learning algorithms for edge detection forces the rotation invariance by creating a dataset of rotated images [25].

2.2.2 Space probability analysis

In the literature, multiple methods exist to generate closed regions from different partial contours (partial contours) [50–53], but they do not provide any spatial information about the pixels not belonging to a contour. They are used for contour completion whose goal is to connect different contour parts to obtain continuous 1D boundaries and closed regions. Hence, they cannot be used jointly with region-based methods since they do not generate 2D information unless binary regions are produced through segmentation.

Other region-based methods allow to generate probabilistic information in 2D by analyzing the 2D space filled with pixels, but not its edges [23,34,54–56]. Hence, there is a discontinuity between the computer vision methods that detect edge-based information and those that detect spatial-based information.

Therefore, we propose the mathematical innovation of the EM and GF kernels which allow analyzing the space probability of inclusion inside a set of thin partial contours [16,21]. It means that using only thin partial contours, we can analyze the space between those partial contours. This is innovative since no other method exists to perform such task, and that we demonstrated that the EM-based convolution kernels are the only possible kernels that allow such a task [16], with the GF kernels being an improvement over the EM kernels [17]. The reason is that the kernels act as

space integrators and are the only kernels that guaranty conservation of energy [31–33]. Hence, they are the only kernels that guaranty that the inside of a shape has a constant probability value, which is mandatory for inclusion probability analysis [16]. This allows the proposed method to act as a bridge between the edge-based methods and the region-based methods. The method is called “Probability of Inclusion Inside Partial Edges” (PIIPE) and is discussed in more details in Chapter 3.

2.3 Gradient-domain image editing

Gradient-domain image editing (GDIE) was first founded by Perez et al. [57] when they proposed the first Laplacian solver (also called Poisson equation solver). By doing so, they were able to do image blending in the gradient domain, which allowed for seamless Poisson blending [57]. Hence, they opened the door to multiple applications for image editing software, such as the suite of GDIE applications proposed in the GradientShop [58]. However, all the applications remain for image or video post-processing and special effect, without any unsupervised application. This is an important aspect of the current thesis since we developed the first method of merging salient object detection with edge detection using the gradient-domain [18].

2.3.1 Poisson solver for seamless blending

The fundamental step of any GDIE-based algorithm is solving the Laplacian (or Poisson equation), which can then be used for applications such as seamless blending and gradient removal [57–59]. Perez et al. proposed a numerical solver for the Laplacian of an image by iteratively minimizing the variational problem [57]. This allowed many others to develop a more optimal method to reduce the computation speed and error [57,60] while others proposed an alternative solver based on the Jacobi method [59]. Although these methods converge with no error, they are hard to implement and require heavy computing since they need iterative computing to solve it. An alternative method of modifying the Poisson problem is proposed by Tanaka [61], but it can only be used for seamless blending since it cannot reconstruct an image from its Laplacian.

2.3.2 Green’s function

The proposed Green function convolution (GFC) method uses the GF in 2D space to solve the Laplacian or Gradient in a single convolution [17]. It is based on the mathematical principal of GF

developed in the 1830s, which are used in mathematics and physics to solve a different kind of inhomogeneous linear differential equation [33]. Using GFC, it becomes possible to solve any Laplacian in n -dimensions by convolving the differential equation with the hypersurface-normalized GF potential [17,33]. To our knowledge, no other computer vision method uses GF for gradient-domain editing or feature detection.

Hence, the method proposed in the current thesis unlocked many possibilities for gradient-domain editing, since it is more precise, faster and easy to implement on a central processing unit (CPU) or a graphics processing unit (GPU) [17]. Furthermore, it was demonstrated using a mathematical proof that the proposed GFC method yields always to the least-error solution [17]. Since the GFC method is both fast and optimal, then it positions itself as an important innovation for the subfield of gradient-domain image editing (GDIE), as discussed in Chapter 5. The current thesis also proposes the first GDIE method for machine learning applications [18,19].

2.4 Feature detection

To build smart systems based on computer vision, one of the most important tasks is to detect the features of an image [23,34,46]. Those features allow defining “interesting” parts of an image according to abstract concepts such as edges and saliency. Edges allow finding the boundary between different regions and objects [50,51], while saliency allows finding the important region in an image, which is often the foreground and the object of focus [26,62]. They are binary problems since their goal is to assign a value of true or false to every pixel, although in practice they assign a value in the range $[0, 1]$ before applying a threshold. This section will present a critical review of the methods used to detect these features. Some of those methods are based on CNN with more details in section “2.5 The convolutional neural network”.

2.4.1 Edge detection

Solving the binary problem of edge detection has seen a great amount of progress in recent years. The edges are the boundaries between distinct image regions, which differ in terms of color, texture, contrast, etc.

The most basic method was to compute the gradient of the image using numerical derivatives [35] and Gaussian derivative filters [41,49]. Then, Canny’s algorithm in 1986 proposed multi-stage

thresholding with edge continuity [63]. Other methods were developed to compute the edges of more complex or noisy images such as watershed models [28], the gravity models [14] and the gPb method in 2010, which combined multi-scale Gaussian derivatives with statistical K-means clustering [49]. To better compare different methods, datasets were created on which to perform rigorous benchmarking of the results, such as the BSDS500 [49], on which 7 different humans traced the “true” edges of the image.

The creation of the datasets started a new revolution in the edge detection world since they could also be used for training machine learning algorithms. One of the first machine learning algorithms is proposed in 2013 and is based on a structured forest [64,65]. Then, from 2015 to 2018, multiple different types of convolutional neural networks (CNN) were developed which outcompeted any previous method both in precision and computation time, such as HED[66], RCF [25] and UCF [11]. Hence, since the year 2015, most edge detection methods are based on deep learning.

2.4.2 Salient object detection

The development of salient object detection (SOD) had similar progress as the edge detection, except that it started with the individual extraction of different statistical features. A salient object is defined as the object of importance in an image, which is often different from surrounding in terms of contrast, color, texture, orientation... It is easier to understand it with the example provided in Figure 2-1 where we observe that the salient object is the person and its clothing. We also observe a major performance difference between the DRFI method based on structured forest [67] and the DSS method based on CNN [26].

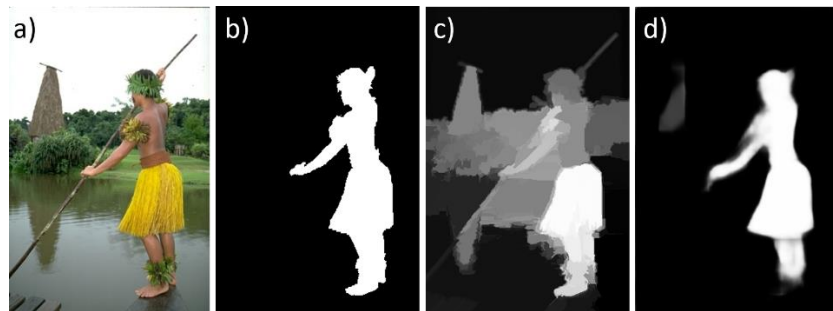


Figure 2-1 : Example of saliency for a person image from the ECSSD dataset [68]; (a) Original image; (b) Ground-truth saliency map; (c) Saliency map produced by the DRFI method [67]; (d) Saliency map produced by the DSS method [26].

Saliency maps were generated by computing statistics based on clustering and density [69–71], concavity [72], contrast filtering [73], backgroundness [74]... Then, the method DRFI [67] based on deep forest is proposed [67]. However, just like for edge detection, it was soon outpaced by the arrival of CNN around 2015. Since then, many CNN-based algorithms are proposed for salient object detection such as DHS [75], DCL [76] and DSS [11,26]. Hence, since the year 2015, there is no incentive to develop any SOD method that is not based on CNN. Details on the CNN architecture of those methods is given in section “2.5.2 CNN architectures for feature detection”.

Seeing how the CNN were easy to use, to train and highly versatile, a recent 2018 work by Hou et al. proposed to create a unified framework architecture UCF for saliency, edge detection and skeleton extraction [11]. This work showed that all 3 features were extracted with more precision than the competing algorithms. This is due to the unique architecture of the UCF, which allows a horizontal cascade of learned features at each scale [11].

2.4.3 Saliency enhancement

There have been numerous research to improve saliency and edge detection algorithms, but so far only a few propose using edges to improve saliency maps [77–79]. There are methods to improve saliency maps using background detection [74], contrast enhancement and texture smoothing [80]. However, Patel et Raman [77] benchmarked these methods and realized that they do not work well on most recent CNN-based models since the deep networks perform better at detecting those features. The BGOF method proposed by Patel et Raman [77] and the denseCRF method [71] were the only ones that demonstratively improved the performance of the newest CNN-based saliency detection [77]. Their work proved conclusive since they optimize the saliency inside the segmented boundaries and reduce it outside the boundary. However, the gain of using their method is minimal and slow to compute, since they minimize an energy function in segmented regions instead of using a real boundary for the improvement.

The proposed approach called saliency enhancement using edges (SEE) proposes to use the gradient-domain (discussed in section 2.3) to merge state-of-the-art (SoA) edge detection methods with SoA saliency method. This kind of method is unique in the literature and proved to outperform any other method using a combination of gradient-domain pre-processing and post-processing. Also, the resulting improvement proved to be much higher than any competing method, including the one proposed by Patel et Raman [77]. Chapter 6 details the proposed SEE method and how it

outperformed competing saliency enhancement algorithms using GDM. To ensure that our method performs objectively better, we use standard benchmarking parameters defined in Appendix A. Furthermore, Chapter 7 shows how we implemented GIS (a variation of GDM) inside the DSS saliency network, which allowed to improve the testing results further while reducing the convergence time, the overfitting and the sensitivity to noise.

2.5 The convolutional neural network

The convolutional neural network (CNN) is today the best machine learning algorithm for machine vision, since it outcompetes any other algorithm for classification [9], segmentation [81], edge detection [11,25], saliency [11,26] and many other applications. They are biologically inspired since they are based on how the visual cortex works in monkeys [12]. In fact, they are able to mimic far better the human vision than standard neural networks (NN) with far better results in every machine vision category. Since NN are known for decades, they are only presented in Appendix A. The following sub-sections will explain how CNN improved on the previous model and present some of the most recent CNN architectures concerning binary feature detection.

Right now, there is no work in the literature that proposed using GF or EM kernels inside any kind of neural network or at their outputs, which is one of the novel ideas brought forward by the current thesis.

2.5.1 Convolutional neural networks

To overcome the limitations of NN, the CNN was created based on the biology of the frontal cortex [12]. By using convolution kernels instead of scalar weights, the CNN allowed doing both the feature extraction and the NN architecture at the same time thus requiring minimal preprocessing [82]. Therefore, the CNN has only one matrix input for each image channel (meaning 3 input matrices for an RGB image) [9].

The architecture of CNN is similar to the NN architecture, except that the input of each neuron is a matrix and that the weights perform a convolution on the input of each neuron [9]. The convolution kernels allow for feature extraction directly inside the CNN by eliminating all the unnecessary connections between far-away pixels, by increasing the bond between nearby pixels

and by being translation invariant. Usually, the kernels are square and odd-sized, with sizes varying between 1×1 and 7×7 [10,20,26], although they can be larger.

2.5.2 CNN architectures for feature detection

The current thesis focuses more on saliency detection, but advances in that field are closely linked to advances in edge detection, image segmentation, and object classification. This section will first discuss the CNN architectures of classification, followed by the architectures of edge detection and saliency.

2.5.2.1 Architectures for classification

For the task of image classification, one of the most successful models created in 2015 is the VGGnet-16, with an accuracy of 93% for top-5 classification error on the extended ILSVRC dataset containing 2000 classes [10]. The image and channel sizes across the network are presented in Figure 2-2 and the convolution sizes are presented in Figure 2-3. We can observe that the network has 14 convolutional layers with a max pooling at every layer to reduce the size of the image in the deeper parts of the network.

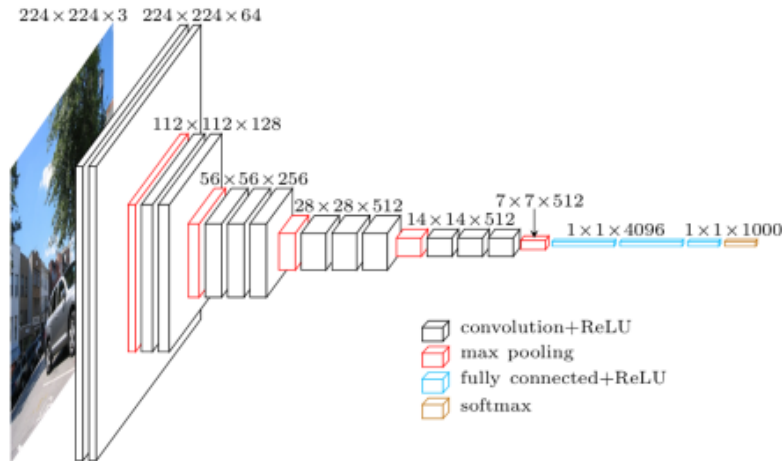


Figure 2-2 : Image and channel sizes throughout the VGGnet-16 given by width × height × channels [10,83]

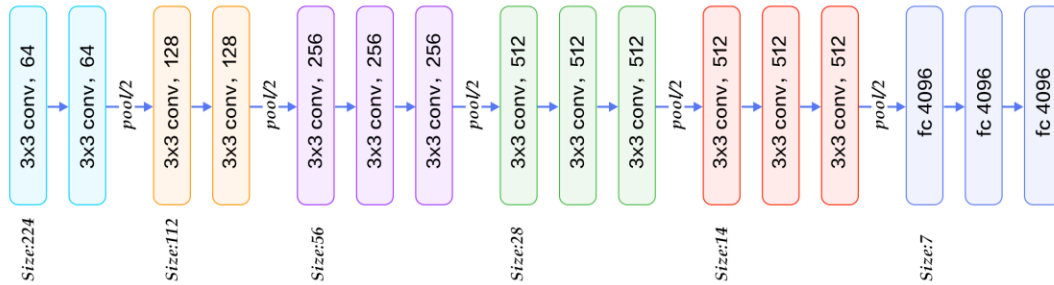


Figure 2-3 : Convolution kernel sizes given by width \times height, number of neurons [10,83]

Other deeper and more performant architectures are used for classification such as InceptionNet (or GoogLeNet) [20] and ResNet [84]. However, the VGGnet-16 is explained in the thesis since it is the standard network used as a nesting for saliency and edge detection [11,26,66,75], although the UFnet method demonstrated that the ResNet performs slightly better for nesting [26].

The InceptionNet has 4 different versions with each version performing better than the previous one. The idea of the network is that *inception modules* can be created such that each module is able to extract a set of features [20]. Then, the InceptionNet appends multiple such modules in a sequence such that their combined feature extraction is further improved [20], as presented in Figure 2-4. Although the InceptionNet v1 performs similarly to VGGnet-16, it reduced the memory requirements by a factor 36, making it more popular [85].

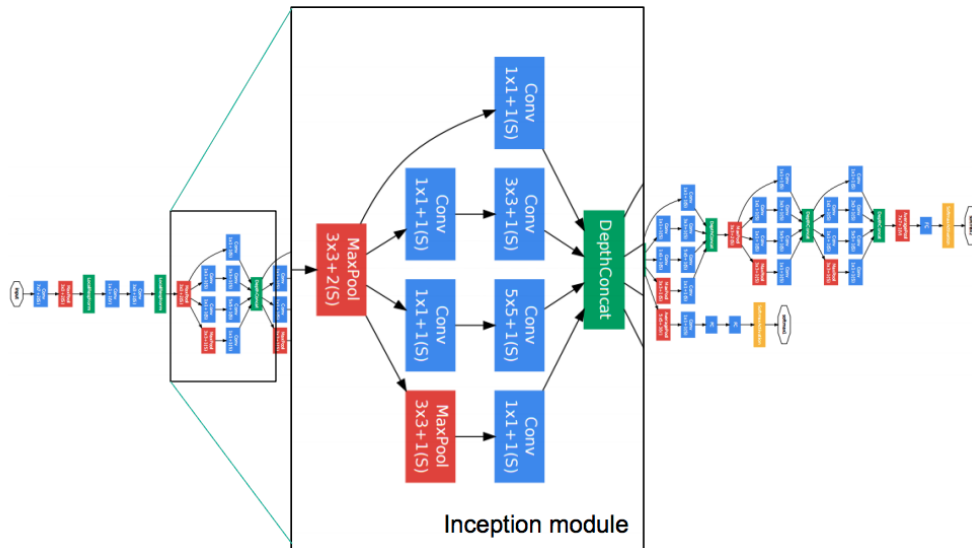


Figure 2-4 : Illustration of the InceptionNet v1 architecture, known as GoogLeNet [20,85].

Following the success of the InceptionNet, the ResNet was developed in 2016, and is based on the idea that abstract image features require very, very deep networks [84]. Hence, they developed an

architecture of 101 and 151 layers [84] presented in Figure 2-5, which is significantly higher than the 16 layers of the previous VGG-net, enabling the ResNet to outperform its competitors by a high margin [84,85].

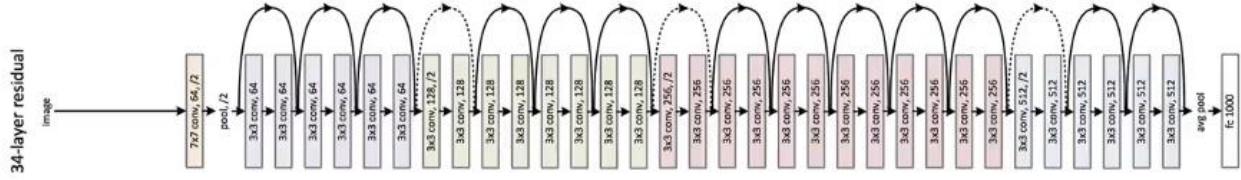


Figure 2-5 : Architecture of the ResNet neural network [84,85]

Modern networks architecture developed between 2016 and 2018 often combine the InceptionNet with the ResNet [86], with a few innovative architectures based on fractals.

2.5.2.2 Nested feature detection

Although feature detection is a different task than classification, the classification networks are heavily optimized to detect complex features in an image. Hence, some of the most successful methods of feature detection, such as HED [66], DHS [75], DSS [26] and UFnet [11] rely on a nested classification network with side outputs connected perpendicular to the classification network. This nesting means that the HED and DSS architectures use side-layers of CNN that are connected at different points inside the VGGnet-16, usually before each max pooling layer. Hence, the HED and DSS methods are able to use a heavily trained and optimized neural network as a nest and focus on training the supplementary side layers for their edge detection or saliency purpose [11,26,66].

2.5.2.3 Saliency using pre-segmentation

Instead of relying on a nested architecture with different upscaling, other methods propose to pre-segment the image into small regions that are likely to have the same saliency values. Such an approach is proposed by the MDF method for saliency detection [87], while the DCL method proposes to use a hybrid between superpixel segmentation and nesting on the VGGnet-16 [76]. Although these methods have an accuracy comparable to the nested algorithms, they are slower to compute and produce non-smooth saliency maps.

2.5.3 Green's function in neural networks

As far as explored, there are no mentions of using either EM or GF inside any type of neural network or machine learning algorithm. Hence, it becomes one of the original ideas proposed by the current thesis.

It seems counterintuitive to think that a GF based convolutional kernel can be used inside a neural network since the CNN optimizes thousands of different convolutional kernels. However, we explained that the kernels usually have sizes varying between 1×1 and 7×7 [10,20,26], which makes it impossible to have an unlimited receptive field and a long-distance interaction between features unless the network is very deep or uses multiple downsizing (such as strided convolutions or strided pooling) [9]. In fact, it is impossible to solve a Laplacian or to project any vector field into its least-error conservative field since they both require a GF kernel at least as big as the image [17]. Hence, we proposed using EM and GF based kernels for CNN applications.

2.6 Summary of the problematic

In summary, there are different lacking's in the literature regarding computer vision and deep-learning algorithms that we try to address in the current thesis. The most important problem is the limitation of the receptive field at high resolution (without downscaling the image). This problem originates from the convolutional kernel with sizes varying between 1×1 and 7×7 [10,20,26]. Firstly, it means that edges cannot interact with regions since they are far away and since edge information cannot extrapolated into regions. Secondly, it means that far-away features cannot interact together to form a cohesive set of features. Finally, it means that it is impossible to make sure that a vector field of features is conservative, meaning that it is smooth and line-integrable.

By developing the EM convolutional kernels and later improve them into the GF convolutional kernel, the current thesis hopes to address many of those problems. In fact, this thesis will propose the first use of GF for gradient domain image editing (GDIE), the first use of GDIE for saliency improvement, the first use of GDIE and GF inside a CNN, and the first use of GF to transform the features of a CNN into conservative features field.

CHAPTER 3 ARTICLE 1: NOVEL CONVOLUTION KERNELS FOR COMPUTER VISION AND SHAPE ANALYSIS BASED ON ELECTROMAGNETISM

Dominique Beaini^a, Sofiane Achiche^a, Yann-Seing Law-Kam Cio^a, Maxime Raison^a _

^aPolytechnique Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

Published on *Arxiv* #1806.07996, June 2018

Abstract

Computer vision is a growing field with a lot of new applications in automation and robotics since it allows the analysis of images and shapes for the generation of numerical or analytical information. One of the most used methods of information extraction is image filtering through convolution kernels, with each kernel specialized for specific applications. The objective of this paper is to present a novel convolution kernel, based on principles of electromagnetic potentials and fields, for general use in computer vision and to demonstrate its use for shape and partial contour analysis. Such filtering possesses unique geometrical properties that can be interpreted using well-understood physics theorems. Therefore, this paper focuses on the development of the electromagnetic kernels and on their application on images for shape and partial contour analysis. It also presents several interesting features of electromagnetic kernels, such as resolution, size and orientation independence, robustness to noise and deformation, unlimited receptive field and the ability to work with 3D images.

Keywords: Shape analysis; Partial contour analysis; Computer vision; Electromagnetic potential field; Feature extraction; Image filtering; Image convolution.

Note to the reader: This chapter is introductory to the thesis and is mainly focused on mathematical introduction and qualitative analysis. Multiple statements are made without rigorous proof, with either a proof in later sections or simply qualitative images as demonstrations. However, the current chapter is still important to understand multiple concepts of the current thesis, such as the conservation of energy, the edges extrapolation using dipoles, the interaction between distant edges, the unlimited receptive field and the resolution invariance. Subsequent chapters are more rigorous in terms of mathematical demonstration and comparison to the state-of-the-art.

3.1 Introduction

Computer vision (CV) is a challenging and interdisciplinary field, with infinite possibilities of images and videos to be processed. Hence, it is not trivial to find the appropriate methodology to extract the desired data from the image. One of the favored approaches for image analysis is convolution kernels, which can be used for blurring, edge detection, defect detection, machine learning, etc. Therefore, creating new convolution kernels could unlock new possibilities of image processing or improve the existing methods.

Choosing the right convolution kernels for a task requires a mathematical understanding of each kernel, and may prove to be a tedious task. To avoid such problems, many methods rely on numerical optimization of the kernels, such as genetic algorithms [88,89] or the highly-praised convolutional neural networks (CNN) [23,34,45]. This usually removes the need to create new types of kernels, since the optimized kernel will often be more efficient than a manually chosen one. However, electromagnetic (EM) kernels possess interesting features that can be used for computer vision, but that cannot be retrieved using standard kernel optimization. The current paper will explain how to build EM kernels, how to apply them and how to extract the useful information of shapes and partial contours.

3.1.1 Related Work

Convolution kernels are one of the favored method for extracting information from an image, and they have been the subject of numerous research works in noise removal [40], defect detection [41,42], image segmentation [29,43], edge detection [14], machine learning [24,44,45], etc. The convolutions allow to quickly scan the whole image to apply local mathematics on nearby pixels, either to extract features or to modify the image [23]. A common characteristic of the convolution kernels of all these methods is that the kernels are usually small in the bi-dimensional plane, with a maximum size of 7×7 [14], 9×9 [41], 11×11 [45], although some physical phenomena require bigger kernels, twice the size of the Image [29,43].

The current paper is not the first one to base its convolutions on natural phenomenon for the purpose of CV. For the biology-inspired techniques, the most prominent one is neural networks (NN) based, more specifically the convolutional neural networks (CNN) [22,23,45]. This technique mimics how the human brain works by combining multiple sequences of neurons for the sake of classification

or learning, and they are currently among the very best techniques for image classification [8,22,24,45].

Some examples of physics inspired techniques for CV include watershed droplets for edge detection [28] and force vector fields used for active contours [29,43], but they are not related to electromagnetism which is presented in this paper. The main methods related to the work presented in this paper are the quadrupole convolution used to define the orientation of contours [13] and the edge detection using gravitational fields (which is mathematically similar to electric fields) [14]. Although these 2 works initially proved to be efficient new ways of analyzing images, they did not realize the full possibilities of using electromagnetism (EM). Therefore, these methods are now outranked by more recent techniques. The current paper aims to go a lot further in exploring the CV possibilities of electromagnetic potentials and fields (EMPF), such as oriented dipoles, interactive segments, and combined potential-field analysis. It aims at developing a methodology of image analysis that directly uses the laws of electromagnetism as described by J.C. Maxwell [30–32].

Furthermore, the current paper deals extensively with the problems of shape and partial contour analysis, which are important in the field of CV [90]. A partial contour is defined as any curve in an image with a single pixel width. Some basic techniques are focused on giving general information about the shapes, such as the perimeter, area, centroid and mean size. These techniques are usually robust to deformations since they use each point of shape for computation, but they do not provide enough information for an advanced analysis [90] since they reduce a 2D shape into a 0D value. Other techniques transform a shape into its contour (either polygonal or smooth) [29,37,38,90,91] or skeleton [90–92], hence transforming a 2D shape into a 1D partial contour. Those partial contours are then analyzed using their local curvatures, intersections or Fourier descriptors [29,37,90–94]. Those techniques provide more information than the 0D values since they reduce the dimensionality by a lower factor, but they still greatly reduce the information initially available in the shape.

3.1.2 Proposed Approach: CAMERA-I

The objective of the proposed approach is divided into 2 parts. The first objective is to develop the mathematics and the methodology required to build and use the EM convolution kernels. The

second objective is to explain how the kernels can be used in CV and to demonstrate their advantages for shape and partial contour analysis.

To answer the first objective, the EM laws are simplified to their static and multi-dimensional interpretation then transformed into convolution kernels. Therefore, the EM potentials and fields of an image can be computed solely with convolutions. By analyzing their values and by determining the attraction or repulsion, it is possible to find several local or global characteristics of the images or shapes. The novel technique proposed in this paper is called “Convolution Approach of Magnetic and Electric Repulsion to Analyze an Image” (CAMERA-I).

For the second objective, this work aims at demonstrating that there are several advantages of the CAMERA-I approach when compared to standard CV methods. One major advantage is the resolution and size independence of the kernels, which is something that is not possible with most other filtering methods [34,36]. Another important characteristic is the high robustness to noise and deformation of the images. In fact, the proposed approach will prove to be way more robust to local changes, since the EMPF will consider the contribution of each pixel in the image, shape or partial contour that is analyzed. Contrarily to standard methods, the proposed EMPF does not reduce the dimensionality of the shapes it analyzes and even allows to analyze 3D objects. Finally, the EMPF approach will show how to extract long distance information about the partial contours and shapes present in an image, allowing to consider the interaction between them and to build a 2D information image from a 1D partial contour.

3.2 Theory of Electromagnetism for Computer Vision

In order for the paper to be self-explanatory, we use this section to remind the reader of the key concepts of electromagnetism (EM) that are useful to understand the paper. It will only deal with the classical theory of EM by J. C. Maxwell [30] and does not deal with relativity or quantum physics. We will then derive some simplified equations of EM, because computer vision deals with a virtual world, and is not subject to conform to the constants of the universe. The laws are modified to only consider the static terms, to remove the constants and to be used in a non-3D world, thus allowing to fully harness the geometrical properties of Maxwell’s equations (MEq) without any physical limitation.

3.2.1 Intuitive exercise

For a better understanding of how electromagnetism can help determine the features of an object, one can consider the physics of a lightning rod. It is well known that lightning will tend to fall on a sharp (highly convex) and tall (far from the center of mass) area, because of the high charge concentration near a sharp point [31]. Consequently, we can determine which regions are concave or convex, and which regions are near or far from the center of mass (CM). It is to note that the current paper goes far beyond this first intuition and uses both mathematical and/or graphical evidence when required.

3.2.2 Electric and Magnetic Monopoles and Dipoles

Electromagnetic charges and dipoles are the key elements behind the CAMERA-I algorithm because parts of the images are treated as EM particles. Such particles generate scalar potentials \mathcal{V} and vector fields \mathcal{E} that will vary according to the distance and the position [30–32]. For the sake of concision, most of the theory is given at “Appendix C.2 Monopoles and Dipoles”. We ignore the fact that magnetic monopoles do not exist, and we suppose that they behave identically to electric monopoles.

3.2.2.1 Electric or magnetic monopoles

The effect of electric monopoles is shown in Figure 3-1 on a normalized color-scale, where \mathcal{V} is represented by the color-scale and \mathcal{E} is represented by the vectors. We see that the positive monopole produces a positive potential and outgoing field, while the negative particle produces a negative potential and an ingoing field.

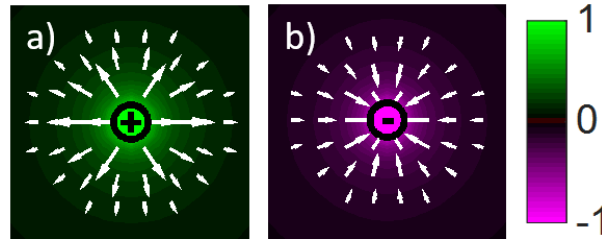


Figure 3-1 : Static electric potential and field of a: (a) positive monopole. (b) negative monopole

When multiple monopoles are put together, then the total potential is given by the scalar sum and the total field is given by the vector sum, as given by equation (1) [30–32].

$$\mathcal{V}_e^{tot} = \sum_i^n \mathcal{V}_e^i, \quad \mathcal{E}_e^{tot} = \sum_i^n \mathcal{E}_e^i \quad (1)$$

3.2.2.2 Electric or magnetic dipoles

When 2 monopoles of opposite signs are placed near each other, it produces a dipole with a potential and field that respect equation (1). Those dipoles can be stacked in several different ways, as observed in Figure 3-2. It is shown that the serial assembling of dipoles does not make it stronger, it only creates a bigger gap between the negative and positive pole. However, placing the poles on 2 parallel lines will create a big dipole with higher potential and field.

Another important characteristic is that when the distance between the poles is small, a dipole in any orientation θ is approximated by equation (2) [31,32], where the superscripts x,y denote the horizontal and vertical orientation of the dipoles. A visual of this superposition is given at Figure 3-2, where it is shown that a horizontal dipole with a vertical dipole is equivalent to 2 dipoles placed at 45° .

$$\mathcal{V}_{dip}^\theta \approx \mathcal{V}_{dip}^x \cos(\theta) + \mathcal{V}_{dip}^y \sin(\theta) \quad (2)$$

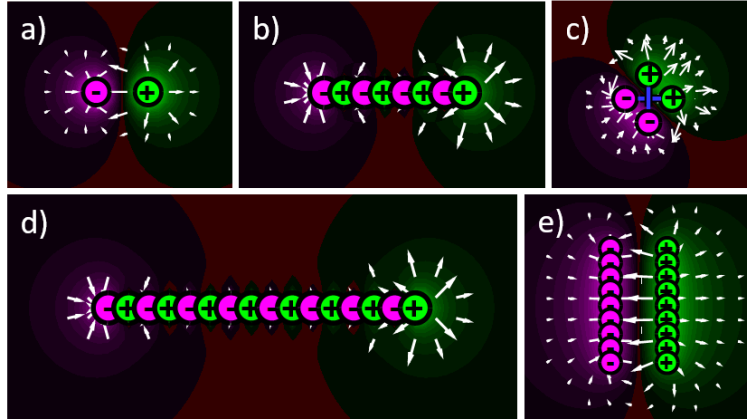


Figure 3-2 : Electric Potential and field for static monopoles placed as (a) a dipole. (b) a small chain of simple dipoles. (c) a horizontal and a vertical dipole, equivalent as 2 dipoles at 45° . (d) a long chain of simple dipoles. (e) simple dipoles in parallel.

3.3 Potential and Fields Equations Adapted for Computer Vision

In order to use the laws of EM, they must first be adapted for computer vision by removing some of the physical constraints and by ignoring the universal constants. In the “Appendix C.3 Mathematical Laws of EM”, Maxwell’s equations are presented and simplified using the assumption that all charges are static. Furthermore, the 4 Maxwell’s equations are reduced to 2 equations, since the electric field equations are symmetric to magnetic field equations. This allows generalizing the potential and field equations in a universe with n spatial dimensions, where n is a real number greater than 1. The modified field is presented at equation (3).

$$\mathbf{E}_{e,m} = q_{e,m} \frac{\hat{\mathbf{r}}}{|\mathbf{r}|^{n-1}}, \quad n \in \mathcal{R}^+ \text{ \& } n \geq 1 \quad (3)$$

By using the electromagnetic laws presented in the appendix, we can write the relation between the potential V and its gradient E at equation (4).

$$\begin{aligned} \mathbf{E}_{e,m} &= -\nabla V_{e,m} \\ V_{e,m} &= -\int_C \mathbf{E}_{e,m} \cdot d\mathbf{l} \end{aligned} \quad (4)$$

It is then possible to determine the potential by calculating the line integral of equation (3). This leads to equation (5), where we purposely omit all the integral constants and where the product constant terms that depend on n .

$$V_{e,m} \propto q_{e,m} \cdot \begin{cases} |\mathbf{r}|^{2-n} & , \quad n \geq 1, \quad n \neq 2 \\ \ln|\mathbf{r}| & , \quad n = 2 \end{cases} \quad (5)$$

For $n = 3$, $V_{e,m} \propto |r|^{-1}$, which is identical to the real electric potential in 3D [30–32]. Because the field is the gradient of the potential, then the vector field will always be perpendicular to the equipotential lines, and its value will be greater when the equipotential lines are closer to each other [31].

For the current paper, the term “**electric**” is used when using monopoles and “**magnetic**” or “**magnetize**” when using dipoles, because it is more intuitive.

3.3.1 Geometrical Interpretation of Potentials and Fields

The mathematical formalism of EM and MEq have been presented and simplified, but with no purpose for shape analysis, which will be the main focus of this section. An in-depth analysis of circles and corners are presented at “C.4 Geometrical Interpretation of Maxwell’s Equations” and can help understand the following interpretation.

If a given shape is filled with positive electric monopoles, then the field will tend to cancel itself near the center of mass (CM) or in concave regions. However, the potential is scalar, which means that it will be higher near the CM or in concave regions. This difference in the behavior of the potential and the field is observed in Figure 3-3.

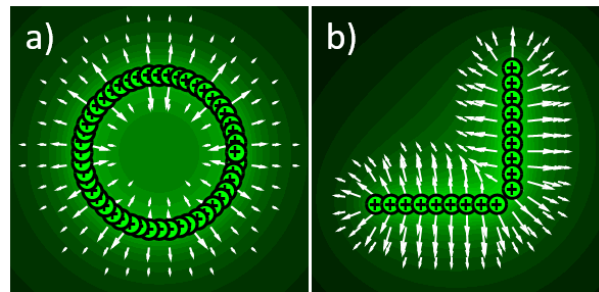








Figure 3-3 : Potential and field with $n = 3$ for positive monopoles placed on (a) A circle. (b) A corner.

Using this difference, we can determine the features of the shape in a given region depending only on the values of V_e or $|\mathbf{E}_e|$. The characteristics of the potential and the field in different regions of the shape are summarized at Table 3.1. Of course, a combination of those factors is possible, like a concave region near the center of mass (CM), which yields to a really high potential and a slightly low field.

Table 3.1 : Potential and Field Characteristics at Different Regions of a Shape Filled with Monopoles, for $n > 2$

Region of interest	Visual	V_e	$ E_e $	Legend
Concave		↑	↓	
Convex		~↓	~↓	↓↓ : really low ↓ : low
Flat		~	↑	~↓ : slightly low ~ : average
Near CM		↑	~	~↑ : slightly high ↑ : high
Far from CM		↓	↓	↑↑ : really high
Inside		↑↑	↓↓	

3.3.2 Convolutions, Potentials and Fields

The equations (4) and (5) are the main equations used in this paper. The potential is first calculated using equation (5) because it represents a scalar, which means it is easy to sum the contribution of every monopole by using 2D convolutions. Then, the vector field is calculated from the gradient of the potential. Convolutions are used because they are fast to compute due to the optimized code in some specialized libraries such as *Matlab*® or *OpenCV*®.

3.3.2.1 Creating the monopole potential kernel

Knowing that the total image potential is calculated from a convolution, the first step is to manually create the potential of a single particle on a discrete grid or matrix. The matrix must be composed of an odd number of elements, which allows having one pixel that represents the center of the matrix. If the size of the image is $N \times M$, it is preferable to have P_e as a matrix of size $(2N + 1) \times (2M + 1)$. This avoids having discontinuities in the potential and its gradient. However, it means that the width and height of the matrix can be of a few hundred elements and take a longer time to compute without efficient libraries. The convolution kernel matrix for P_e is calculated in the same way as V_e at equation (5), because it is the potential of a single charged particle, with the distance r being the Euclidean distance between the middle of the matrix and the current matrix element. An example of a small P_e matrix of size 7×7 is illustrated in Figure 3-4, where it is noted that P_e is forced to “1” at the center for continuity purpose.

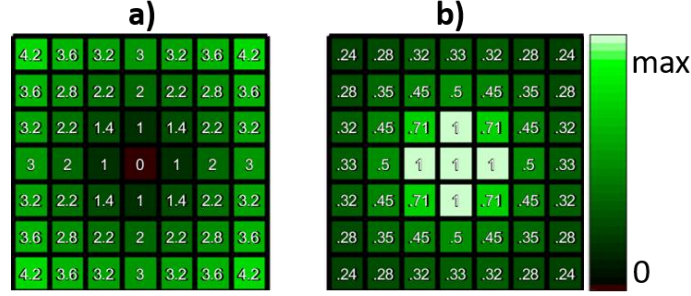


Figure 3-4 : Example of convolution kernel for a particle potential matrix P_e of size 7×7 ;
 (a) Euclidian distance from center r . (b) Potential of a centered monopole $P_e = V_e$, $n = 3$.

3.3.2.2 Creating the dipole potential kernel

Convolutions with dipole potentials can be used also to create an anti-symmetric potential and find the specific position of a point. Therefore, it is required to create a potential convolution kernel for a dipole P_{dip} . We have to remember that a dipole is simply 2 opposite monopoles at a small distance from each other. This can be expressed as a mathematical convolution where P_{dip} is given by equation (6), and is visually shown in Figure 3-5. If divided by a factor 2, we can notice that this convolution is similar to a horizontal numerical derivative (shown later at equations (8) and (9)), meaning that the dipole potential is twice the derivative of the monopole potential [31].

$$P_{dip}^x = P_e * [-1 \quad 0 \quad 1], \quad P_{dip}^y = -(P_{dip}^x)^T \quad (6)$$

$$\text{size}(P_{dip}) = \text{size}(P_e)$$

Using equation (2) along with equation (6), it is possible to determine equation (7), which gives the dipole kernel at any angle θ .

$$P_{dip}^\theta \approx P_{dip}^x \cos(\theta) + i P_{dip}^y \sin(\theta) \quad (7)$$

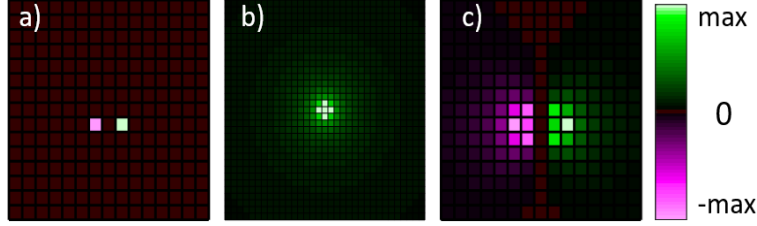


Figure 3-5 : Steps to calculate the normalized potential kernel for a dipole (a) Positive and negative monopoles at 1 pixel distance. (b) Potential kernel P_e . (c) Dipole potential kernel P_{dip}^x resulting from the convolution of image “a” with kernel “b”.

3.3.2.3 Creating the derivative kernels

Derivative kernels are important to compute the field because we know from equation (4) that the field $\mathbf{E}_{e,m}$ is the gradient of the potentials $V_{e,m}$. To use the numerical central derivatives, we simply need to apply the convolution given at equation (8), with the central finite difference coefficients given at equation (9) for an order of accuracy (OA) of value 2 [95].

$$\frac{df}{dx} \approx f * \Delta^x, \quad \frac{df}{dy} \approx f * \Delta^y \quad (8)$$

$$\Delta^x = (\Delta^y)^T = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}, \quad \text{OA} = 2 \quad (9)$$

3.3.2.4 Calculating the potential and the field of an image

A crucial step for the CAMERA-I technique is to transform an image into charged particles, which will allow calculating the electric potential and field. The first step is to determine the position and intensity of the charge. Each pixel with value +1 is a positive monopole, each pixel with value -1 is a negative monopole, and each pixel with value 0 is empty space. Therefore, the pixels of the image represent the density of charge and have values in the interval $[-1, \dots, 1]$, where non-integers are less intense charges.

Next, the P_e matrix is constructed as seen on Figure 3-4, and applied on the image with the convolution shown at equation (10). Then, the horizontal and vertical derivatives are calculated using equation (8) and give the results for E^x and E^y . Finally, the norm and the direction of the field are calculated using equation (11). It is possible to visualize these steps at Figure 3-6, where a quadrupole is represented.

$$V_e = I * P_e, \quad \text{size}(V_e) = \text{size}(I) \quad (10)$$

$$E^{x,y} = V_e * \Delta^{x,y}$$

$$|E| = \sqrt{(E^x)^2 + (E^y)^2} \quad (11)$$

$$\theta_E = \text{atan2}(E^y, E^x)$$

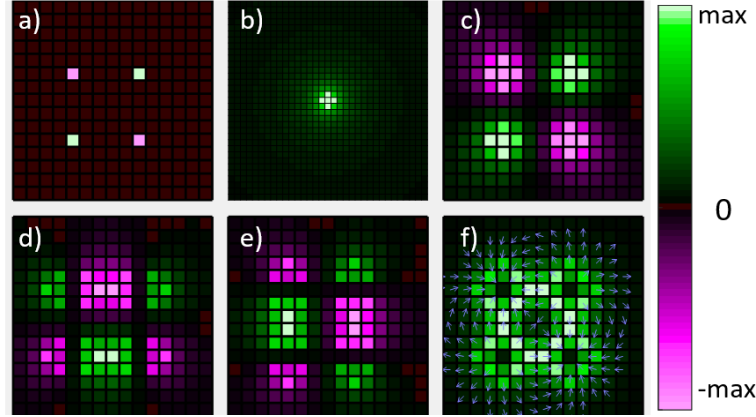


Figure 3-6 : Calculation of the potential and field of an image (a) Monopoles in the image. (b) Potential kernel P_e . (c) Total potential V_e . (d) Horizontal field E_e^x . (e) Vertical field E_e^y . (f) Field norm $|E_e|$ and direction

The same process that is used to transform each pixel into a monopole can be used to transform them into a magnetic dipole, by using the result presented in Figure 3-5 as the kernel. The steps and results are shown in Figure 3-7 where each pixel is transformed into a horizontal magnetic dipole with $\theta = 0$. The formula to calculate the magnetic potential using a convolution is given at equation (13), with the density correction factor F shown at equation (12). This density factor F allows to consider the fact that pixels placed in diagonal have a lower number of pixels per unit length then those placed horizontally. The angle θ depends on the image, as it is often chosen to be either parallel or perpendicular to the direction of the element to magnetize. Also, the matrix size of V_m is the same as the matrix size of I . The real part is chosen in equation (13) to represent dipoles perpendicular to θ , while the imaginary part represents dipoles parallel to it.

$$F = \max(|\cos(\theta)|, |\sin(\theta)|)^{-1} \Rightarrow 1 \leq F \leq \sqrt{2} \quad (12)$$

$$V_m = \Re \left((I \circ F \circ e^{i\theta}) * P_{\text{dip}}^\theta \right) \quad (13)$$

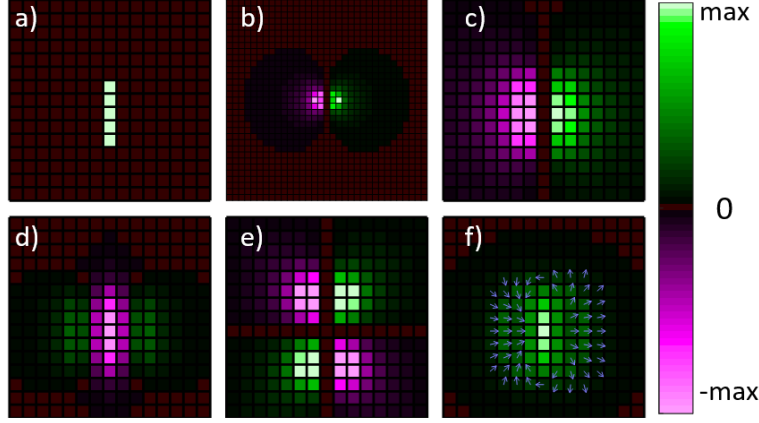


Figure 3-7 : Steps to calculate the magnetic PF of an image (a) Dipoles in the image. (b) Horizontal dipole potential kernel P_m^x . (c) Total potential V_m . (d) Horizontal field E_m^x . (e) Vertical field E_m^y . (f) Field norm $|\mathbf{E}_m|$ and direction.

It is to note that the image (e) of Figure 3-7 is really similar to the image of quadrupole potential presented in Figure 3-6. This is because it represents 2 consecutive perpendicular derivatives of the potential of monopoles $\partial/\partial x (\partial/\partial y V_e)$, which is mathematically equivalent to a quadrupole potential.

3.4 Application of EM Convolutions

The previous section explained how to correctly build the convolution kernels, although real kernels are a lot bigger than the schematic demonstrations. In this section, the focus will shift on how to use those kernels for shape and partial contour analysis, and what are the advantages of using EM convolution kernels.

3.4.1 Detecting Shape Characteristics

This first sub-section will focus on the use of EM convolution kernels for the detection of multiple shape characteristics, such as the convex or concave regions, and the relative distance to the centroid.

3.4.1.1 Finding the regions of interest

It was discussed in the section “2.4 Geometrical Interpretation of Potentials and Fields” that the electric potential and field can be used for shape analysis, with a summary of the characteristics presented at Table 3.1. To demonstrate those characteristics, a special shape is created with all the mentioned regions of interest (RoI), with the computed potential and field shown at Figure 3-8.

The index “onC” means that the values were set to 0 everywhere but on the contours. It is to note that the PF are computed using the whole surface of the shapes and that the values are set to 0 after the computation of V_e and $|\mathbf{E}_e|$. The contour of a full shape can be easily determined with morphological operations. The values of the potential V_e^{onC} and field $|\mathbf{E}_e^{\text{onC}}|$ on the contours are squared to show a better contrast between the low values and the high values. They are also thickened using image dilation, for the purpose of showing better images.

The value of the dimension is set to $n = 3$ for these examples, as it is found experimentally to be ideal for such an analysis. By choosing a value of $2 < n < 3$, a similar interpretation can be done, but it will increase the contribution of the pixels very far from each other, and significantly reduce the contribution of nearby pixels. By choosing a value of $n > 3$, it will reduce the contribution of pixels that are far from each other, and increase the contribution of nearby pixels. Hence, the value of $n = 3$ was found to be a good equilibrium of the contribution of nearby and far pixels, although each specific application could optimize its value. For a more advanced analysis, it is possible to use various different dimension values, such as $n = \{2.3, 3, 4\}$, and to combine the information given by each value of n .

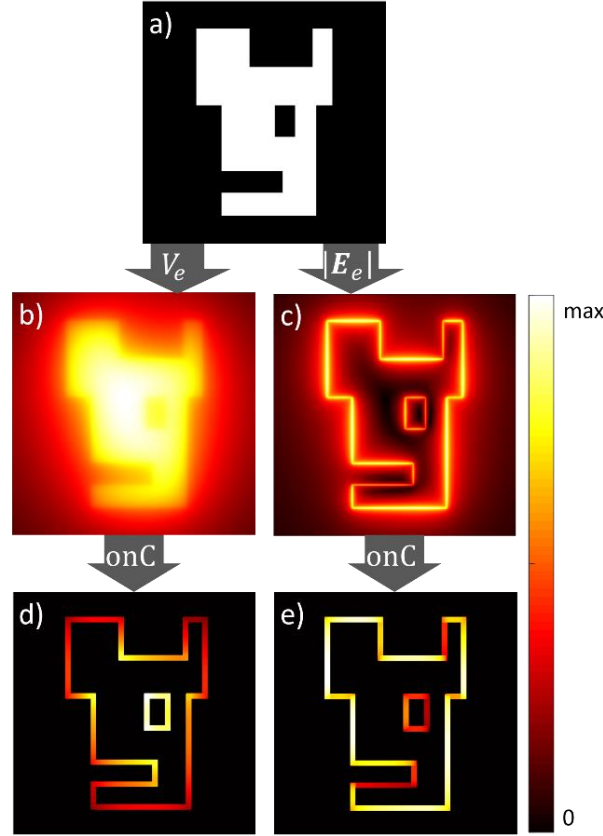


Figure 3-8 : (a) Special shape with the white region being a uniform density of charge, used to compute the following PF with $n = 3$. (b) The potential V_e . (c) The field $|E_e|$. (d) The potential squared only on the contour $(V_e^{\text{onC}})^2$. (e) The field squared only on the contour $|E_e^{\text{onC}}|^2$.

Using the values of V_e^{onC} and $|E_e^{\text{onC}}|$ depicted at Figure 3-8, it is possible to find the regions of interests, as seen at Figure 3-9. The percentile thresholds that are used are shown in Table 3.2. Since the shape that is used is complex, the regions are not perfectly discernable, as usually expected. For example, a concave region (which expects a high value of V_e) can also be far from the CM (which expects a low value for V_e), hence, the thresholds are contradictory. However, this can be used as an advantage, since it allows to use general information about an image, and make it more robust to noise. In fact, regular convolution kernels are small, which makes them vulnerable to small variations in the shapes contours, but it is not the case for EM kernels. EM kernels are also invariant in rotation and robust to deformations.

It is to note that using constant thresholds might lead to problems regarding the continuity of a region, which could be fragmented in a few small parts. To avoid this problem, all regions are grown by a security factor, which is chosen as 5% of the biggest dimension of the shape (this factor

can be changed depending on the needs). Using such a percentage allows the growing to be robust, no matter the resolution or the size of the shape. The algorithm for such a region growing is explained in Algorithm 10-A, at the “Appendix C.5 Partial contour Manipulations”.

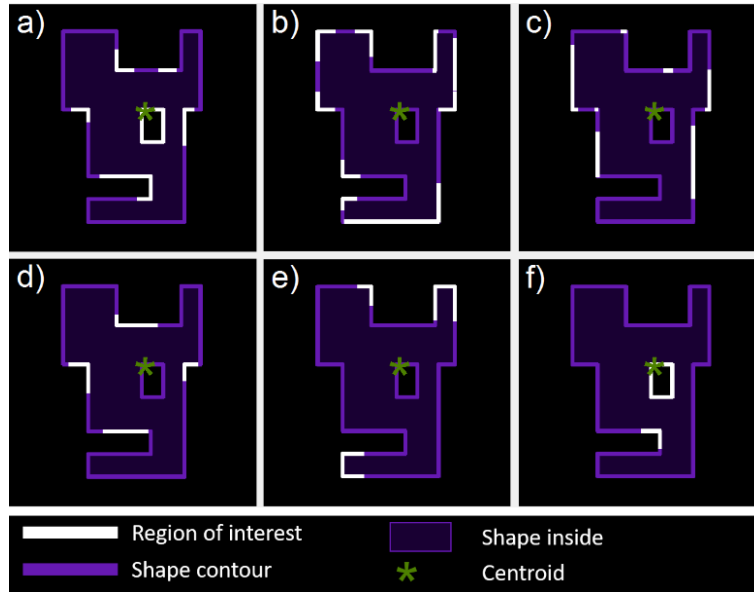


Figure 3-9 : RoI found on a complex shape using a contour analysis by potential and field thresholds. (a) Concave regions. (b) Convex regions. (c) Flat regions. (d) Regions near the CM. (e) Regions far from the CM. (f) Regions inside the shape.

Table 3.2 : Percentile Thresholds Used for the Discovery of the Regions of Interest

Region of interest	Thresholds percentile for V_e		Thresholds percentile for $ E_e $	
	Min (%)	Max (%)	Min (%)	Max (%)
Concave	70	100	0	50
Convex	15	40	15	40
Flat	40	60	80	95
Near CM	80	95	40	60
Far from CM	0	25	0	25
Inside	90	100	0	10

3.4.1.2 Robustness to deformation

To demonstrate the robustness of the technique, the shape of Figure 3-9 is modified using a combination of the following filtering, both on small and big scale: twirl, twist and wave. The shape resulting from the filtering is presented at Figure 3-10, with the RoI computed using once again the thresholds of Table 3.2. As observed, the discovered regions for both Figure 3-9 and

Figure 3-10 are almost identical, with only minor differences. From all the RoI, the only differences comprise of one convex region, one flat region, and one region far from the CM. All other regions are present on both figures at the same place. Those differences are minor and are expected since the shape has been greatly modified by the multiple filtering.

Hence, we show that the proposed technique is highly robust against shape and contour deformation for detecting RoI. This is mainly due to the electric field that considers every pixel inside a shape, not only those in a small region of the contour. In fact, although the contour of the shapes in Figure 3-9 and Figure 3-10 are greatly different, the total pixels inside the shape area had a lot less variation, which means that the values of the EMPF are almost identical. Other convolution kernels are small, meaning that they focus only on local information. Hence, kernels that detect concave regions will detect any bump in the contour that is locally concave, making it really vulnerable to deformation.

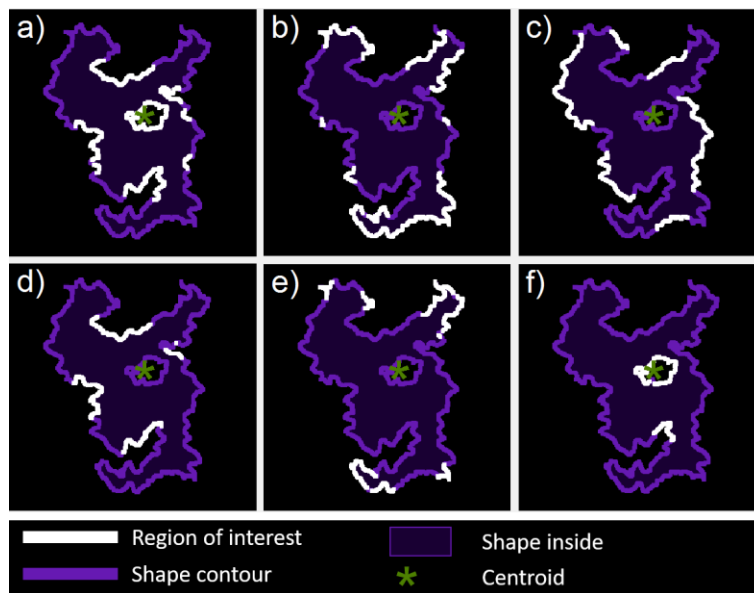


Figure 3-10 : RoI found on a complex shape (filtered with a twirl, twist and wave distortion) using a contour analysis by potential and field thresholds. (a) Concave regions. (b) Convex regions. (c) Flat regions. (d) Regions near the CM. (e) Regions far from the CM. (f) Regions inside the shape.

3.4.1.3 Analysis of 3D shapes

In spite of being robust to deformation, an important characteristic of EMPF is that they can be used in 3D without any added complexity. Of course, computation time will be longer in 3D, but it is partly compensated by resolution which is usually lower than 2D images.

However, since there is more information in 3D than 2D, it is easier to analyze 3D objects using different values of n . The same rules of Table 3.1 still apply, but a value of $n < 4$ will be more sensible to the CM, while a value of $n > 4$ will be more sensible to the local convexity. An example of result for a 3D mug is presented at Figure 3-11, with $n = \{3, 4\}$. It can be observed that $|E_{onc}|_{n=3}$ is better at determining the inside of a cup with opposing faces, while $|E_{onc}|_{n=4}$ is better at finding the bottom of the cup, where the concavity is the highest. Furthermore, $(V_{onc})_{n=4}$ is better than $(V_{onc})_{n=3}$ at finding the local convexities at the border of the cup.

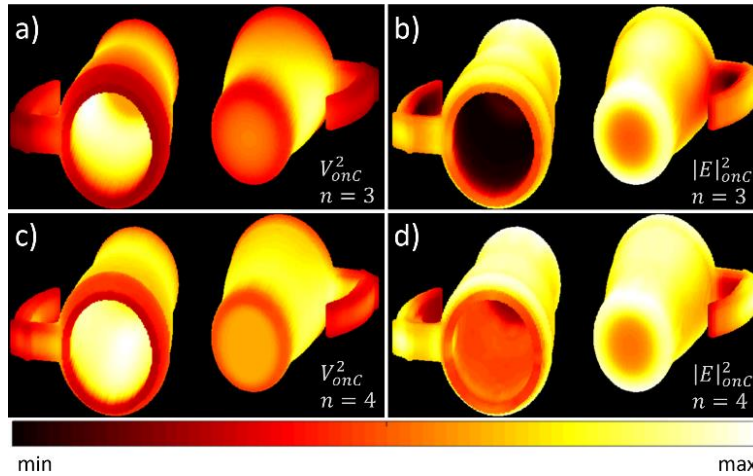


Figure 3-11 : EM potential V and field E generated by a 3D mug, with different values of n . (a)

V^2_{onc} with $n = 3$. (b) $|E|^2_{onc}$ with $n = 3$. (c) V^2_{onc} with $n = 4$. (d) $|E|^2_{onc}$ with $n = 4$.

3.4.2 Magnetic Repulsion for Partial contour Interaction

As demonstrated in the last section, the electric potential and field allow analyzing a shape and its contour. In this section, a new tool will be developed to show how magnetism can be used to analyze thin partial contours and their interactions. A thin partial contour is defined as any curve or line that has only one-pixel width. Hence, each pixel of the partial contour has a maximum of 2 neighbors, except at the intersection of multiple partial contours.

3.4.2.1 Choosing the Magnetic Dimension

One major difference between analyzing full shapes and partial contours is the impact of image resolution. For a full shape, if the resolution is lowered, the total relative area between the shape and the image remains the same.

For a thin partial contour, if the resolution is lowered, then each pixel of the partial contour is wider. Hence the partial contour has a bigger relative area when the resolution is low. This causes problems when using EM convolutions since the area represents the total charge. However, it is found that using $n = 2$ for the EMPF makes it invariant of the thickness of the partial contour and the resolution of the picture. This can be observed at Figure 3-12, where 2 partial contours of different resolutions are magnetized perpendicular to the partial contour with dimensions $n = 2$ and $n = 3$. For the value of $n = 2$, presented at the subfigures (c) and (f), we can observe that the equipotential lines are exactly the same. However, this is not the case for subfigures (b) and (e), where $n = 3$. The potentials of Figure 3-12 are computed using equation (13), with θ being the orientation perpendicular to the line. To find the orientation θ for any kind of partial contour, it is possible to use Algorithm 10-B in the “Appendix C.5 Partial contour Manipulations”.

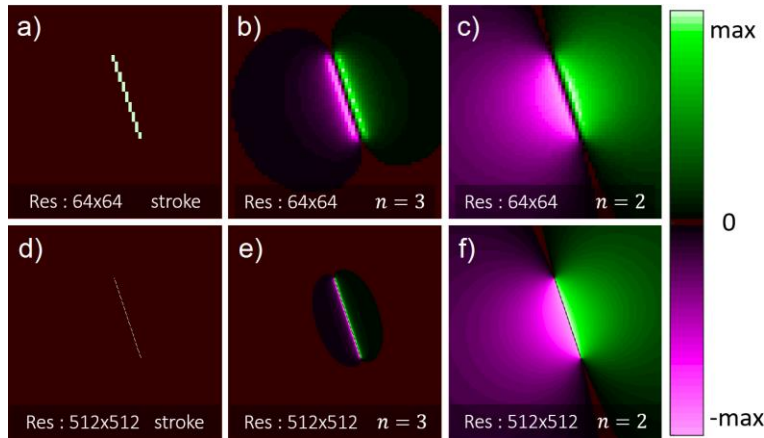


Figure 3-12 : Potential V_m resulting of the convolution of a dipole perpendicular to the partial contour lines. (a) Partial contour with low resolution 64x64. (b) Dipole with $n = 3$ and low resolution. (c) Dipole with $n = 2$ and low resolution. (d) Partial contour with high resolution 512x512. (e) Dipole with $n = 3$ and high resolution. (f) Dipole with $n = 2$ and high resolution.

Another important characteristic of the potential with $n = 2$ is that it is the only dimension which ensure a conservation of energy in the potential and field of the image, since the image is in 2D. In fact, the conservation of energy is the reason why a thin partial contour requires $n = 2$ to be

invariant of the image resolution. If we chose a value of $n > 2$, then some energy will be lost in the higher dimensions as we go further from the EM particles. Inversely, a value of $n < 2$ will create more energy as we go further from the EM particles. Using the same principles, it is possible to deduct that a thin plane in a 3D image requires $n = 3$ to be invariant of the resolution.

The conservation of energy means that Gauss's Theorem can be applied to the field produced by a partial contour. By using Gauss's Theorem, we can know that any closed partial contour, which is magnetized perpendicular to its direction, will produce a null field both inside and outside the partial contour. The fact that the field is null means that the potential is constant, both inside and outside the partial contour, but with different values. This can be observed in Figure 3-13, where the closed partial contour is chosen to be a circle. The more the circle is near closing, the more the potential is uniform. However, this is only true for $n = 2$. The value of V_m is given by equation (13), with θ computed using Algorithm 10-B.

In summary, the dimension value for the partial contour analysis must be $n = 2$, for both purposes of resolution invariance and conservation of energy.

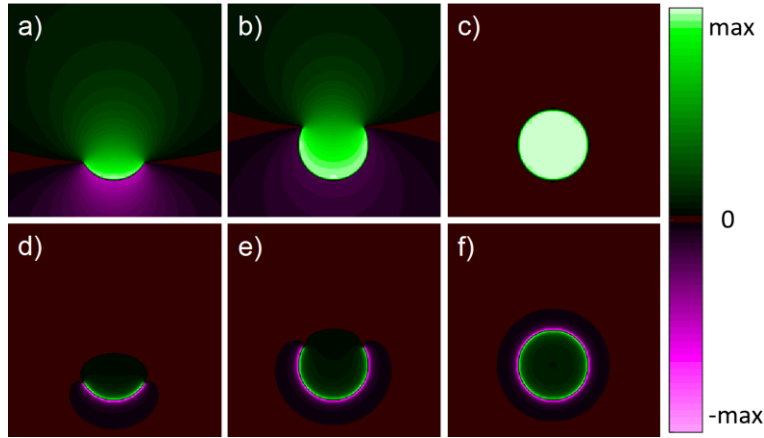


Figure 3-13 : Potential V_m of a circular partial contour magnetized perpendicular to their orientations. (a) Circle arc of 90° , with $n = 2$. (b) Circle arc of 270° , with $n = 2$. (c) Circle arc of 360° , with $n = 2$. (d) Circle arc of 90° , with $n = 3$. (e) Circle arc of 270° , with $n = 3$. (f) Circle arc of 360° , with $n = 3$.

3.4.2.2 Magnetic Interaction

As seen previously in Figure 3-13 with $n = 2$, a partial contour that is almost closed will have a higher potential $|V_m|$ inside it, with a lower potential outside. This can also be applied to 2 partial

contours that interact with each other by magnetizing them perpendicular to the partial contours with equation (13). It is possible to shift the value of θ by a factor of π on each partial contour to flip the positive and negative side. By choosing carefully which partial contour is flipped, it is possible to maximize the magnetic repulsion in an image, as shown at Figure 3-14.

When there is a magnetic attraction, which is when the positive (green) part of a partial contour meets the negative (pink) part of another partial contour, nothing interesting happens in terms of the potential. However, when there is a repulsion (positive meets positive, or negative meets negative), there is a high concentration of potential $|V_m|$ between the partial contours, with an almost constant value (low field $|E_m|$). Henceforth, the magnetic interaction is interesting, as it offers an opportunity to analyze the whole 2D space using only thin 1D partial contours in the initial image.

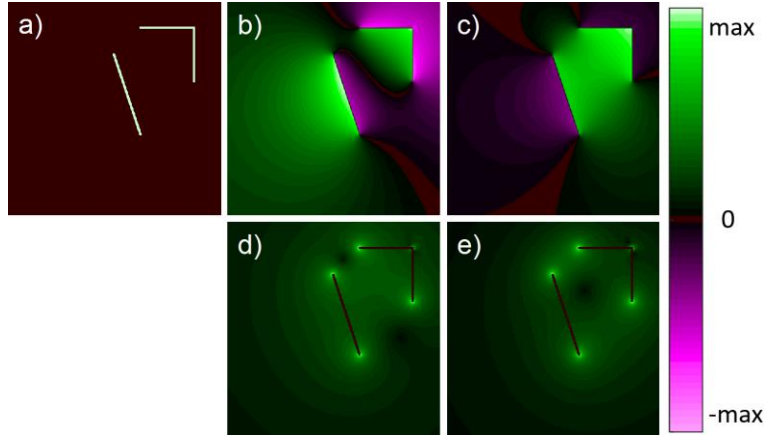


Figure 3-14 : PF computed from the initial partial contour, with $n = 2$ and the dipole perpendicular to the partial contours. (a) Initial partial contour. (b) Potential of attraction V_m . (c) Potential of repulsion V_m . (d) Field of attraction $|E_m|^{0.5}$. (e) Field of repulsion $|E_m|^{0.5}$.

3.4.2.3 Partial contour Analysis

Similarly to the problem of detecting shape characteristics using electric PF, presented in the section “3.1 Detecting Shape Characteristics”, it is possible to detect the characteristics of a partial contour using magnetic PF. Furthermore, the partial contour analysis will be robust to deformation, for the same reasons as the robustness of the shape analysis. To analyze a partial contour, one must simply consider the potential $|V_m|$ produced by dipoles placed perpendicular to the partial contour, using equation (13) and Algorithm 10-B. Then, as seen on Figure 3-13, a concave region will produce a higher value of $|V_m|$, while a convex region will produce a lower value. This is also

analogous to the magnetic repulsion interaction presented in the section “3.2.2 Magnetic Interaction”. An example of this method and its robustness is presented in the Figure 3-15, where it is observed that the values of $|V_m|^2$ are almost identical for the partial contour, the deformed partial contour and the heavily distorted partial contour.

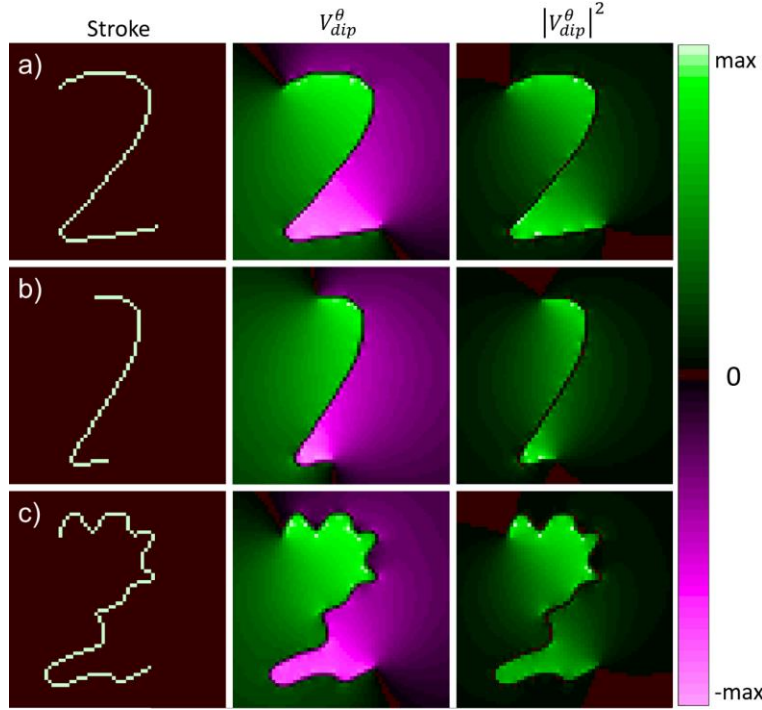


Figure 3-15 : Partial contours for the number “2” at the left, with the potentials V_m of dipoles perpendicular to the partial contours, with $n = 2$. (a) Clean partial contour. (b) Deformed partial contour. (c) Heavily distorted partial contour.

3.4.3 Summary of The Advantages of Electromagnetic Convolution Kernels

The focus of the current paper was mostly about the development and the appropriate usage of EM convolution kernels. Although many characteristics were presented, no concrete application is developed, making it harder to understand the real advantage of such an unusual approach. Hence, this section will focus on enumerating and explaining the great advantages and the uniqueness of EM for image analysis, when compared to other methods.

3.4.3.1 Resolution and Size Independence

The first clear advantage is the resolution and size independence of any EM convolution kernel. This characteristic is unique and is present both in shape and partial contour analysis. This means

that no matter the resolution of the image or the size of the shapes or partial contour in the image, the ideal kernel size is $(2N + 1) \times (2M + 1)$, with N and M being the width and height of the image. If the resolution of the image is doubled, the kernel size is also doubled, but it won't change the results (although a super low resolution will be prone to numerical errors and might change the characteristics of the image). This is a characteristic that most kernels in the literature do not respect, since they are typically with a size between 3×3 and 31×31 [34,36]. Hence, changing the resolution of the image or the elements inside the image requires to change the resolution of the kernels [34,36]. This is problematic since the pixel width and height of a feature is unknown and is not necessarily dependent on the resolution of the image.

3.4.3.2 Orientation independence

In addition to being independent of size and resolution, the proposed EM kernels are also orientation independent, meaning that any rotation applied to the image will not alter the results. This is a feature that is usually available only with rotation symmetric kernels [34], such as Gaussian filters or the P_e kernel presented at equation (5). However, the current method also presents how to use the asymmetric kernel P_{dip}^θ of equation (7) such that it is independent of the orientation. This is because the value of θ is dependent on the local orientation of a partial contour, which changes along with the rotation of the image. Hence, what is presented is a unique asymmetric kernel that is independent in size, resolution and orientation. This is in contrast with other kernels, such as texture algorithms, which usually require 4 to 11 scales and 2 to 8 orientations [34,96], for a total of 4 to 88 filters required for the same feature detection.

3.4.3.3 Robustness to Deformation and Noise

Another interesting feature is the robustness to heavy deformation, which was previously shown in Figure 3-10 and Figure 3-15. This is something that standard kernels cannot handle well due to their size. In fact, a small kernel will be way more affected by a local distortion, meaning that a standard kernel will find convex and concave regions almost everywhere in the presented distorted figures [96]. Furthermore, the standard techniques of contour approximation, such as the polygonal approximations and the Fourier descriptors, are too heavily affected by heavy variations on the contour [90,93]. This is because those techniques rely only on the pixel of the contours, while the approach considers every pixel inside the shape. Those pixels inside the shape are far less affected

by deformation than those on the contours. Similarly, adding noise to the pixels inside a shape will negligibly affect the total potential and field generated, since positive and negative noises will tend to even out.

3.4.3.4 Unlimited receptive field

An important feature of EM kernels is that they also allow taking into account the interaction between different partial contours or shapes, as shown in Figure 3-14. It potentially allows to group multiple partial contours together, or to find the total PF generated by multiple shapes. This is a characteristic that is only possible using big kernels since it is impossible for a small kernel to link 2 distant partial contours. Other methods also propose an unlimited receptive field, such as the force vector fields for active contours [29,43], as they use vector kernel to find the force interaction between each part of a contour.

3.4.3.5 Full Space Information

Another unique feature of the EM kernels is that they allow analyzing pixels that are not in the shape or partial contour of interest. For example, the partial contour analysis of Figure 3-15 allows telling if each pixel is positioned inside the concave regions of the number “2”, or if it is in the convex region. This is impressive since convolution kernels usually give only local information of an image. However, in that case, the EM kernels are able to generate 2D information from a 1D partial contour.

3.4.3.6 Does not Require Shape Approximations

The EMPF approach has the unique capability of not reducing the dimensionality of the studied shapes. In fact, it was stated in the section “1.1 Related Work” that the other shape analysis techniques reduce the shape into a 0D value or a 1D contour/skeleton. These dimensionality reductions make the analysis simpler, but they tend to remove some critical information. Furthermore, techniques such as polygon approximation and Fourier descriptors require to approximate the shape of an object, which does not work well with complex shapes or shapes with holes [37,38,90].

Furthermore, EMPF even has the possibility of being used on 3D shapes, as seen in Figure 3-11, since the laws of EM can still be applied using equation (5). Since 3D shapes are far more complex

than 2D shapes, the advantage of using the proposed CAMERA-I approach is even greater, as it does not require any shape approximation.

3.4.3.7 Cannot be Learned

Since the dawn of CNN's, there is no point of creating a new convolution kernel if it can be learned by the network. The reason is that CNN's use dozens or hundreds of optimized kernels [45], meaning that any useful and “learnable” kernel will be obtained by the network optimization. However, the EM kernels presented in the current paper cannot be learned by such methods, and for several reasons. First, it was already mentioned that the kernels in a CNN are small [22,23,45], usually less than 11×11 . Hence, it is impossible to learn a kernel that is twice the size of the image. Another important reason is the use of the magnetic potential V_m that requires to convert the image into complex numbers using Euler's formula $\exp(i\theta)$, convoluted with a kernel of complex values, as seen in equation (13), with the angle θ being related to the direction of the partial contour. This kind of specific feature is impossible to generate throughout the optimization of standard CNN, since they do not use complex numbers.

3.4.4 Comparison with the literature

The advantages presented in the previous section highlighted the interesting characteristics of the CAMERA-I approach for Computer Vision. A summary of these advantages is listed in Table 3.3, with a direct comparison to state-of-the-art methods of image analysis.

Table 3.3 : Qualitative Comparison Between Different Image Analysis Methods

Characteristics	CAMERA-I	CNN's [23,34,45]	Fourier Descriptors [37,38]
Resolution and size independence	✓		✓
Orientation independence	✓		✓
Robustness to deformation	✓		✓
Unlimited receptive field	✓		
Full space information	✓	✓	
Allows shape analysis	✓	✓	✓
Allows partial contour analysis	✓	✓	
Can be adapted to any problem		✓	
Can be easily used for machine learning		✓	
Does not require heavy computing	✓		✓

From Table 3.3 one can see that the CAMERA-I approach is complementary to CNN's, but in direct competition with Fourier descriptors. One needs to note that the examples of results using Fourier descriptors with 4 or 32 harmonics are illustrated in Figure 3-16. First of all, we can clearly see that 4 harmonics is not enough to describe complex shapes. Using 32 harmonics, the results look better, but there is a lot of oscillations, which makes it difficult to accurately determine the convex and concave regions since this technique relies on the local curvature. In addition, we can observe that Fourier descriptors cannot deal with holes in the shapes, and must consider the holes as separate shapes, contrarily to the CAMERA-I approach (as previously shown in Figure 3-9 and Figure 3-10). Finally, Fourier descriptors are only good to analyze full shapes, and cannot be used for partial contour analysis or partial contour interactions, which is another advantage of the proposed approach.

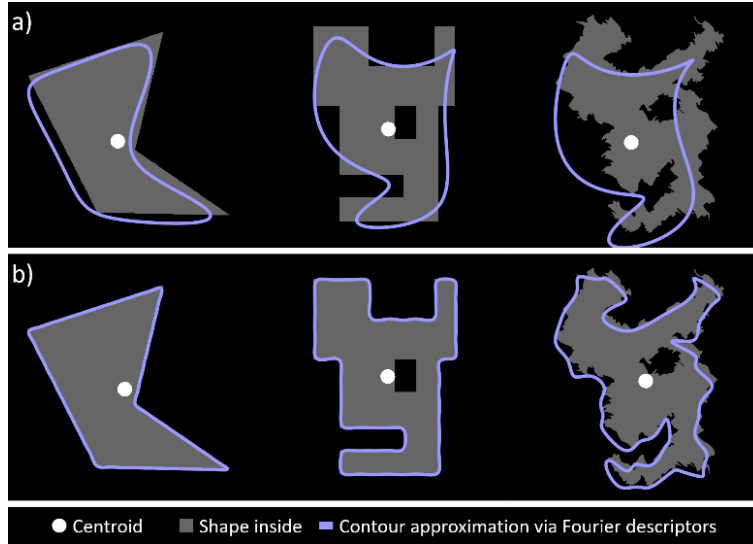


Figure 3-16 : Contour approximation via Fourier descriptors. (a) Fourier descriptor with 4 harmonics. (b) Fourier descriptors with 32 harmonics. [37,38]

3.5 Conclusion

The objective of this paper was to develop different electromagnetic convolution kernels that can be used in computer vision applications and to demonstrate its effectiveness for shape and partial contour analysis. The paper showed how to express the images as electromagnetic particles with possible varying density, allowing to efficiently compute the potential V_e and field $|\mathbf{E}_e|$ associated to them, using convolutions. Using the computed values, it is possible to quickly determine some regions of interest, such as the convex or concave regions, and the proximity to the centroid. This

method was demonstrated to be robust to noise and heavy deformation, and invariant to size, resolution and orientation.

Furthermore, a novel directional magnetic convolution is presented at equation (13), which allows computing a V_m and field $|\mathbf{E}_m|$ that depend on the local density and orientation of thin partial contours. This is a unique way of applying convolution kernels, which proved to be robust to heavy deformations, to allow high distance partial contour interaction and to determine local or global partial contour characteristics. Plus, it offers a unique way to analyze all the pixels in a 2D image depending on their relative position to the 1D partial contour.

In summary, the electromagnetic kernels proved to be an efficient and robust way to analyze images, with unique characteristics that make it impossible to be the result of an optimized or learned kernel. In fact, the EM kernels proved to be independent of the image size, resolution or orientation, in addition to being really robust to deformation. A continuation of this work could focus on the development of specific applications based on those properties.

Acknowledgment

We would like to thank NSERC, through the discovery grant program, and FRQNT/INTER for their financial support as well as MEDITIS (Biomedical technologies training program) through NSERC (FONCER) initiative.

CHAPTER 4 ARTICLE 2: COMPUTING THE SPATIAL PROBABILITY OF INCLUSION INSIDE PARTIAL CONTOURS FOR COMPUTER VISION APPLICATIONS

Dominique Beaini^a, Sofiane Achiche^a, Fabrice Nonez^b, Maxime Raison^a

^aPolytechnique Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

^bUniversity of Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

Published on *Arxiv* #1806.01339, June 2018

Abstract

In Computer Vision, edge detection is one of the favored approaches for feature and object detection in images since it provides information about their objects' boundaries. Other region-based approaches use probabilistic analysis such as clustering and Markov random fields. In fact, only image segmentation can produce regions based on edges, but it requires thresholding by simply separating the regions into binary in-out information. Hence, there is currently a gap between edge-based and region-based algorithms, since edges cannot be used to study the properties of a region and vice versa. The objective of this paper is to present a novel spatial probability analysis that allows determining the probability of inclusion inside a set of partial contours. To answer this objective, we developed a new approach that uses electromagnetic convolutions and repulsion optimization to compute the required probabilities. Hence, it becomes possible to generate a continuous space of probability based only on the edge information, thus bridging the gap between the edge-based methods and the region-based methods. The developed method is consistent with the fundamental properties of inclusion probabilities and its results are validated by comparing an image with the probability-based estimation given by our algorithm. The method can also be generalized to take into consideration the intensity of the edges or to be used for 3D shapes. This is the first documented method that allows computing a space of probability based on interacting edges, which opens the path to broader applications such as image segmentation and contour completion.

Keywords: Computer vision; Partial contour; Probability of inclusion; Edge interaction; Image convolution; Electromagnetic potential field.

Definitions and acronyms

Path : A function of time $S(t)$ that starts at position $S(t_i) = \gamma_i$ and ends at position $S(t_f) = \gamma_f$

Contour : A closed path with only 1 intersection at points $S(t_i) = S(t_f)$

Partial contour : Part of a contour, for time $t_i \leq t \leq t_f$

Edge : Weight associated with the probability that a given pixel is at the boundary of 2 regions

CAMERA-I : Convolution Approach of Magnetic and Electric Repulsion to Analyse an Image

PIIPE : Probability of Inclusion Inside Partial Edges

4.1 Introduction

Image analysis and understanding is a challenging subject in computer vision since there is an infinity of different images and videos that can be processed. Hence, properly extracting information from an image is a difficult task that often requires heavy computation and complex methodologies [23,34]. One possible approach for image analysis is using probabilistic algorithms that allow comparing different parts of an image with their respective characteristics, which can be used for texture understanding [97,98], image segmentation and clustering [55,56,99] and machine learning [100]. They are also used by several researchers for probabilistic image construction based on Markov fields or deep learning [101–103], allowing to fill parts of the images that are missing and generate artificial images.

One distinction between the cited algorithms is that edge-based methods generate information in a 1D space composed of thin edges [23,28,34,64], while the region-based methods generate information in a 2D space composed of pixels [23,34,54–56]. Currently, multiple existing methods group edges to generate closed regions [50–53], but they do not provide any spatial information about the pixels not belonging to a contour. This implies that they cannot be used jointly with other region-based methods. Hence, there is a need to develop a novel probabilistic algorithm that generates spatial information based on the edges of an image, since it will close a gap in image analysis and could therefore unlock new possibilities. The approach proposed in this paper differs from any other existing algorithm since it provides spatial information based only on thin edges, a unique feature that does not exist elsewhere in the literature. This feature can then be used in

different computer vision algorithms, such as contour completion [50–53] and edge-based image segmentation [104,105] or saliency [106,107].

In this paper, 4 similar concepts are used, namely a *path*, a *partial contour*, a *contour*, and an *edge*. It is therefore important to fully understand the distinction between them. The full definitions are given in “Definitions and acronyms”, with the time t used to define the progression of the parametric functions, where t_i is the initial time and t_f is the final time. In summary, a path is any function $S(t)$, a contour is any non-self-intersecting closed path, a partial contour is any partial contour, and an edge is a weight associated to a pixel present at the boundary of 2 regions. An example of those concepts is presented in Figure 4-1.

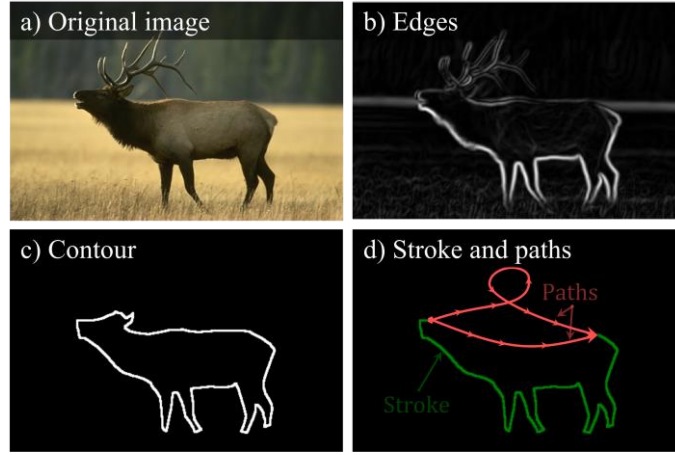


Figure 4-1 : Definitions of different concepts. (a) Image of an elk from BSD500 dataset. (b) Edges computed using the Sobel algorithm. (c) The contour of the elk. (d) Partial contour (stroke) along with 2 possible paths that close the partial contour.

In our previous research work [15], we reported that electromagnetic (EM) convolutions allow analyzing different properties of a shape or a partial contour. We demonstrated how the EM dipoles can be chosen to be invariant in regards to the size, the resolution and the orientation of a partial contour, thus allowing its analysis. Also, it was confirmed that the EM kernels are robust to distortions and deformation [15,21], which makes them ideal for the analysis of the general behavior of a complex partial contour. Furthermore, we showed that the EM approach allows generating information in the whole 2D space, based only on the 1D partial contour. This allowed us to take into consideration the interaction between different partial contours, their general concavity and to analyze the space between different partial contours [15]. Improving the

algorithms for partial contour analysis can be useful in multiple applications, such as shape analysis [92,108], object discovery [6,109,110] and object grasping [38,44].

Building upon our previous research [15], the objective of the research work presented in this paper is to develop a new and improved method for computing the space probability of inclusion inside a partial contour using dipole electromagnetic convolutions [15], with the assumptions that any partial contour is meant to be closed and that different partial contours interact together. This paper will emphasis on developing the algorithm, but it will not present any application apart from the images used for exemplifying the mathematical concepts. Hence, it is the precursor of future application-focused work. The main objective is reached by completing the following steps:

Determine an analytical representation for computing the probability of being included inside partial contours, using a finite set of possible curves.

Generalize the results for a continuous space of probability using an uncountable set of circular curves.

Study the characteristics of the probabilities to ensure their consistency.

Demonstrate the equivalence between the space probabilities of step “2” and the computation of numerical magnetic convolutions.

Develop the algorithm to compute the space probability on complex images, where multiple shapes and contours are present.

The validation of the developed method will be carried out by showing how the partial contours can be used to generate an estimation of the original image which was used for edge detection [63,64]. The approach is based on the premise that each edge should form a closed contour and uses this premise to compute the probability that each point in space is contained within the given contours. Hence, based only on their shape and their position, it can determine the regions of interaction and the partial contours that do not belong together. Thus, it differs fundamentally from any other probabilistic method in computer vision since it does not need information about color, texture, intensity, motion, etc.

The proposed technique is called PIPE for Probability of Inclusion Inside Partial Edges, and it belongs to the general approach CAMERA-I [15,21] (Convolution Approach of Magnetic and

Electric Repulsion to Analyze an Image) developed in our laboratory at École Polytechnique de Montréal. Hence, the full name of the approach is CAMERA-I-PIIPE.

4.2 Computing the inclusion probabilities with circular paths

This section aims at understanding how to compute the probabilities that any point is enclosed within an open partial contour, knowing that a single path should close the partial contour. First, this section will justify that circular paths have the ideal characteristics for enclosing paths. Secondly, it will show how an infinite number of circular paths can be used to compute the probability of enclosure. Finally, the properties of the computed probabilities and their validity are analyzed.

4.2.1 The importance of subsets regions

This subsection presents the concept of computing the probability of inclusion for a partial contour, which requires to consider different possible paths that close the given partial contour. Although the most trivial path between 2 points at the extremities of the partial contour is a straight line, the developed technique requires to consider different possible paths for the computation of the space of probabilities. This is because a single path to close the partial contour will lead to only 2 possible values being “0” (outside the contour) and “1” (inside the contour). Hence, a space of probabilities other than “0” and “1” requires more possible paths.

To generate simple and intuitive paths, the paths between 2 points should be non-self-intersecting, convex and smooth, as discussed in more details in the appendix “D.2.1 Characteristics of the paths between 2 points”. Then, it is possible to define a path S_n that passes by the extremities $\gamma_{i,f}$ of a given partial contour S . Therefore, if S and S_n do not intersect, it is then possible to define a region R_n which is bounded by S and S_n . This is shown in Figure 4-2, where the region R_n contains the point γ_{in} but excludes the point γ_{out} . A more rigorous definition of R_n will be given at section “4.2.3 Intersecting circular arcs” in equation (22).

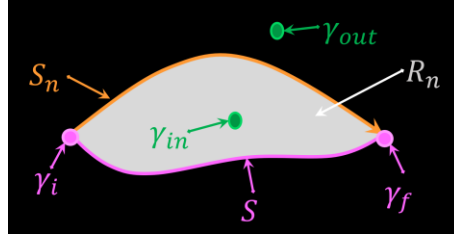


Figure 4-2 : Example of a partial contour S between points $\gamma_{i,f}$, closed by a path S_n to generate the region R_n containing the point γ_{in} but excluding γ_{out} . S is an existing partial contour and does not have restrictions. S_n is the generated path used to close S , thus S_n must be non-self-intersecting, convex and smooth.

The probability that a given point γ_{in} is inside the region R_n can be computed if we allow a finite number of regions N_R that are partially bounded by S , where there is a smaller number N_γ of regions R_n that contain γ_{in} ($\gamma_{in} \subset R_n$), versus the total number of regions N_R . Then, by assuming that each path S_n is equiprobable, it is possible to compute the probability P_S of being inside the partial contour S using equation (14).

$$P_S(\gamma \subset R_n) = \frac{N_\gamma}{N_R} \quad (14)$$

To compute the probabilities given by (14), it is required to find the values of N_γ and N_R . To significantly reduce the complexity of the problem, we can choose the paths S_n such that it does not intersect any other path $S_{m \neq n}$, except at the points γ_i and γ_f (noted $\gamma_{i,f}$). We also define S_n^+ and S_n^- , with each sign representing a path on a different side of S . The numbering variable n^\pm and the angle β_n^\pm are also defined according to the sign and numbering of S_n^\pm .

Therefore, if we suppose that a path S_n^\pm does not intersect $S_{m \neq n}^\pm$, that it is associated to a starting angle β_n^\pm (refer to Figure 4-3), and that each angle β_n^\pm is smaller than the next angle β_{n+1}^\pm , then we can deduce that each region R_n^\pm will be a subset of the region R_{n-1}^\pm . This relation is expressed in equation (15), with an arbitrary example presented in Figure 4-3 using $n^+ = [1, \dots, 5]$ and $n^- = [1, 2]$.

$$\left. \begin{array}{l} S_n^\pm(t_n) \neq S_{m \neq n}^\pm(t_m) \forall \{t_n, t_m\} \\ \text{AND} \\ \beta_n^\pm < \beta_{n+1}^\pm \end{array} \right\} \Rightarrow R_n^\pm \subset R_{n-1}^\pm \quad (15)$$

Since the angles β^+ and β^- have the same starting and ending points but in different directions, then the relationship between them is given by equation (16).

$$\beta^- = 2\pi - \beta^+ \quad (16)$$

For any non-infinite value of N_R , the value of N_γ depends of the value of n that respects the condition $\gamma \subset R_n$. For example in Figure 4-3 where $N_R = 7$, the probability P_S of any point being within the region R_n can be computed using equation (14), with the result shown in equation (17).

$$P_{S_{example}}(\gamma \subset R_n^\pm) = \frac{N_\gamma}{N_R} = \frac{n^\pm}{N_R} = \frac{n^\pm}{7} \quad (17)$$

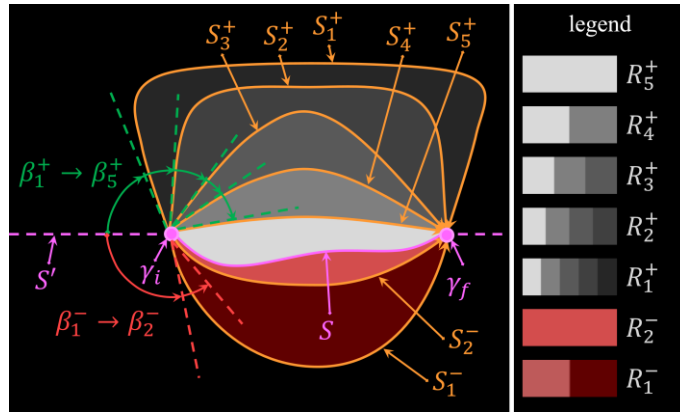


Figure 4-3 : Example of 7 paths S_n between points γ_i and γ_f , with starting angles $\beta_{1 \rightarrow 5}^+$ and $\beta_{1 \rightarrow 2}^-$, such that R_n , the region between S_n and S , is a subset of R_{n+1}

It is worth noting that since every region R_n^\pm is a subset of R_{n-1}^\pm , it is possible to compute the probabilities P_S of belonging to the region R_n in the case of a finite set of partial contours using equation (14). Still, it is even more important in the case of an uncountable set of partial contours, since it will allow generating a continuous space of probabilities. To generate such an uncountable set of partial contours, one can define a partial contour $S(\beta^\pm)$ for any angle $\beta^\pm = [0, 2\pi]$. Hence, there will be an infinite number of regions, meaning that the ratio in equation (14) will yield an indetermination. However, since there is a single curve S^\pm associated to each angle, and since the regions R_n^\pm are subsets of R_{n-1}^\pm , then the indetermination can be solved by replacing N_R by the total span of β^\pm , and N_γ by the span of β^\pm such that $\gamma \subset R_n^\pm$. Therefore, the probabilities P_S can be

computed using equation (18), where β_γ^\pm is the biggest angle that contains the point γ . Since β_γ^\pm is bounded by 0 and 2π , then the probability is also bounded by the inequality (19).

$$P_S(\gamma \in R^\pm) = \frac{\text{range}(\beta^\pm(\gamma \in R^\pm))}{\text{range}(\beta^\pm)} = \frac{\beta_\gamma^\pm}{2\pi} \quad (18)$$

$$0 \leq P_S \leq 1 \quad (19)$$

4.2.2 Circular paths between 2 points

The previous section showed that it is possible to compute P_S using equation (18) for an uncountable set of paths, without explaining how to generate such a set. Hence, this section will present how to generate a set using circular paths. Circular paths are ideal since they are smooth C^∞ , convex, symmetric and non-self-intersecting. Also, the set of circles passing by 2 constant points cover the entire 2D space, as discussed in more details in the appendix “D.2.2 Choosing the circle, rejecting the parabola”.

An example of such a circular path S_C is given at Figure 4-4, where the only independent variables are β and x_0 , with β being the starting angle and x_0 being the half-distance between the points $\gamma_{i,f}$. All the other variables, such as the radius, the area and the height of the circle, are dependent variables with the equations given in the appendix “D.2.3 Circular path parameters”. The Cartesian equation of the circle is given at (20), where the radius is $x_0 \csc \beta$, and the vertical offset is $x_0 \cot \beta$.

Let us note that the circle resulting from the angle β^+ is the same as the one resulting from the angle $\beta^- = \pi - \beta^+$, with $S_C^+(\beta^+)$ associated to one part of the circle, and $S_C^-(\beta^-)$ associated to the complementary part of the same circle (see Figure 4-4). Also, $S(\beta)$ is a set that contains an uncountable number of circles, since each angle β represents a different circle. One could argue that it is easy to generalize the circular equation to an ellipse equation, but it violates the laws of electromagnetism discussed later in section “4.3 Computing the probabilities in an image using EM”, as explained in more details in the appendix “Elliptical potentials and paths”.

$$x_0^2 \csc^2 \beta = x^2 + (y - x_0 \cot \beta)^2 \quad (20)$$

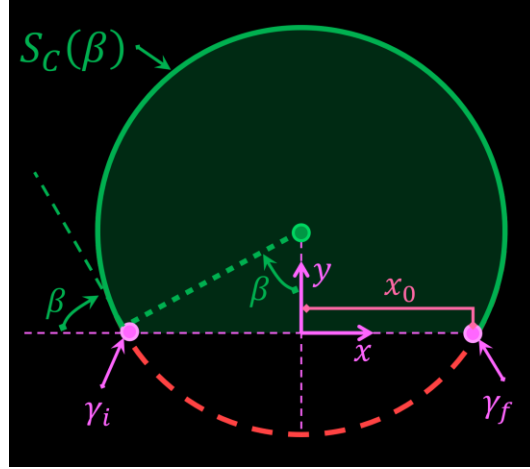


Figure 4-4 : Example of a circular path between points γ_i and γ_f , with a starting angle β

4.2.3 Intersecting circular arcs

The previous section presented the mathematical equations of a circle between 2 points, but it did not deal with the partial contour S that needs to be closed. This section will explain how to take it into consideration, and how to deal with multiple intersections between S and S_C . This will allow to determine the region R for any path S_C and compute the probability P_S for any point.

An example of a path S closed by different circular paths $S_C(\beta_n^+)$ is shown at Figure 4-5, where the point γ^+ is at the boundary of $S_C(\beta_2^+)$ and well contained into $S_C(\beta_1^+)$. In that case, it is simple to compute the probability P_S at any point along $S_C(\beta_{1,2}^+)$ using equation (18).

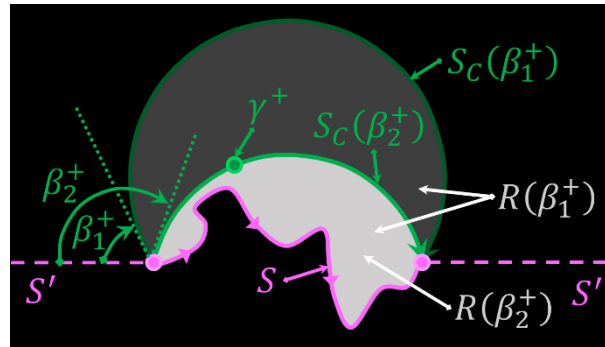


Figure 4-5 : Example of 2 regions $R(\beta_{1,2}^+)$ formed by the closure of the path S with the circular arcs $S_C(\beta_{1,2}^+)$

It becomes more complex to compute P_S when there are intersections between S and S_C at the point γ_\times in Figure 4-6, since it is harder to determine where is the region R . Such intersections will

happen with any partial contour S , except if S is a circle with the same parameters as S_C . Therefore, it is important to be able to deal with such possibilities. In Figure 4-6, we can observe that the region R , which contains both points γ^\pm , can be defined as the region between S and S_C , with $P_S(\gamma^+)$ associated to the angle β^+ and $P_S(\gamma^-)$ associated to the angle β^- . However, such definition does not hold well for non-trivial intersections.

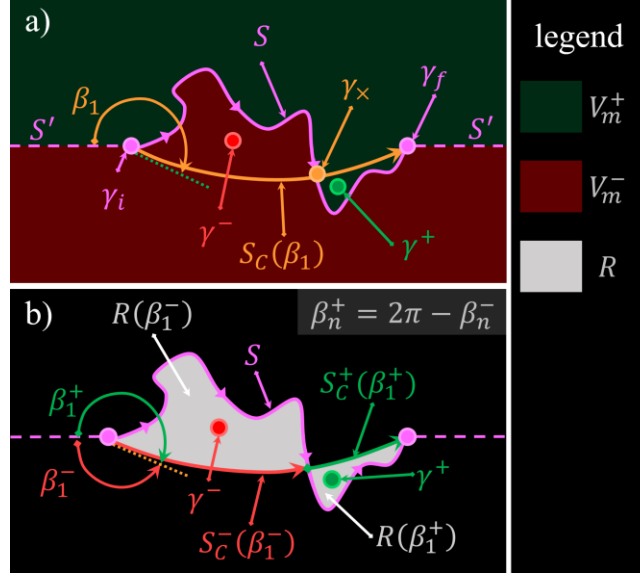


Figure 4-6 : Example of (a) a partial contour S that intersect a circular arc S_C at the point γ_x . (b) The region inside the closure of the partial contour S with the sub-paths S_C^- and S_C^+ .

An example of a complex intersection is given in Figure 4-7, where it is not intuitively clear which region should be counted inside or outside the grayed region R . To solve this problem, let's consider the infinite partial contour $S'_X(t)$ as the continuation of the partial contour S_X along the line $\gamma_i \rightarrow \gamma_f$, as given by equation (21), where S_X represents either S or S_C . Also, to avoid unnecessary complications, we will assume that S'_X is not self-intersecting. In that case, S'_X separates the space in 2 half-spaces. Note that t is the *time* used to represent the parametric equation, with t_i the time associated to $S_X(t_i) = \gamma_i$ and t_f the time associated to $S_X(t_f) = \gamma_f$.

$$S'_X(t) = \begin{cases} S_X(t) & t_i \leq t \leq t_f \\ \frac{t - t_f}{t_i - t_f} \gamma_i + \frac{t - t_i}{t_f - t_i} \gamma_f & \text{otherwise} \end{cases} \quad (21)$$

For the half-spaces generated by S_C , we will define R_C^\pm as the half-space containing $y \rightarrow \pm\infty$. For the half-spaces generated by S , we will define V_m^\pm as the half space containing $y \rightarrow \pm\infty$. Then, the

region R will be defined by the region resulting of the logical equation (22), with an example depicted at Figure 4-7.

$$R = (V_m^+ \cap R_C^-) \cup (V_m^- \cap R_C^+) \quad (22)$$

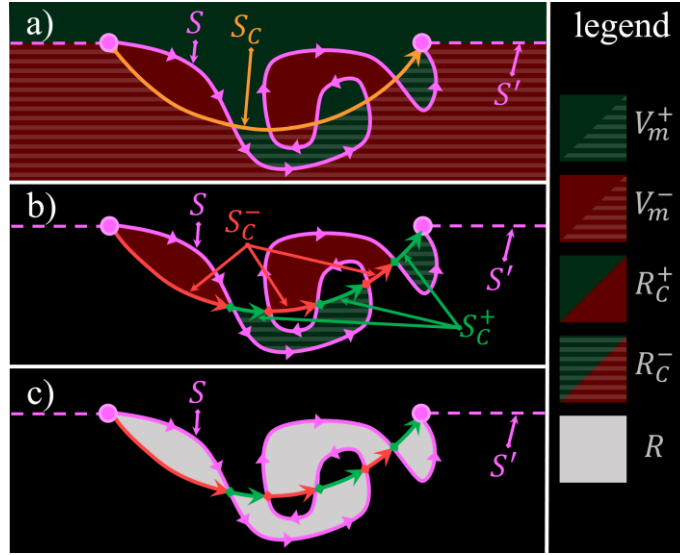


Figure 4-7 : Example of (a) A complex intersection between S and S_C . (b) The regions that do not respect the logical operation are eliminated. (c) The region R is formed with a union of the remaining regions.

Using the inside region definition of equation (22), it is possible to conclude that the probability P_S of any point being inside S is given by equation (18), where β_γ^\pm is the angle that generates the elliptical arc S_C^\pm that passes through the point γ^\pm . Hence, for any point in the region V_m^+ , the value of β^+ is used in equation (18), while for any point in the region V_m^- , the value of β^- is used.

4.2.4 Characteristics of the probabilities

The previous sections showed how to determine which points are inside the region R , and how to compute the probabilities using only the starting angle β . However, they must respect some basic properties in order to be mathematically valid, which will be the main focus of this subsection. It was already demonstrated that equation (18) respects the laws of probabilities with $P_S = [0, 1]$, since it respects the inequality (19). This section focuses on the analysis of other properties, such as certainty of inclusion/exclusion and complementarity, by exploring the mathematical boundaries of the model.

One boundary condition of the proposed mathematical model is that any point γ_∞ infinitely far from S must respect the equation (23). Since the only way for a circular path to reach a point infinitely far is when $\beta = 0$, then using equation (18) with $\beta = 0$ leads to equation (23).

$$P_S(\gamma_\infty) = 0 \quad (23)$$

Other characteristics can be studied at the boundary condition where γ_S is defined as a point infinitely near S . We can choose a point S_i on S , with a vector \vec{v} perpendicular to S at point S_o , as depicted in Figure 4-8 (a). Then, we can define the points γ^\pm as 2 points situated at the opposite side of S at a perpendicular distance, as depicted in Figure 4-8 (a) and in equation (24). The point $\gamma_{S_n}^\pm$ is defined by the mathematical limit when the distance approaches 0 in equation (25). By computing the probabilities of γ_S^+ and using the equations (16) and (18), as seen in equation (26), we can find the property of complementarity presented at equation (27), with some visual examples at Figure 4-8 (b). This complementarity is required for the probabilities to make sense, since it means that the point γ_S^+ is inside S only when γ_S^- is outside S , and vice-versa.

$$\gamma^\pm = S_i \pm \vec{v} \cdot t \quad (24)$$

$$\gamma_{S_i}^\pm = \lim_{t \rightarrow 0} \gamma^\pm \quad (25)$$

$$P_S(\gamma_{S_i}^+) = \frac{\beta^+}{2\pi} = \frac{2\pi - \beta^-}{2\pi} = 1 - P_S(\gamma_{S_i}^-) \quad (26)$$

$$P_S(\gamma_{S_i}^+) + P_S(\gamma_{S_i}^-) = 1 \quad (27)$$

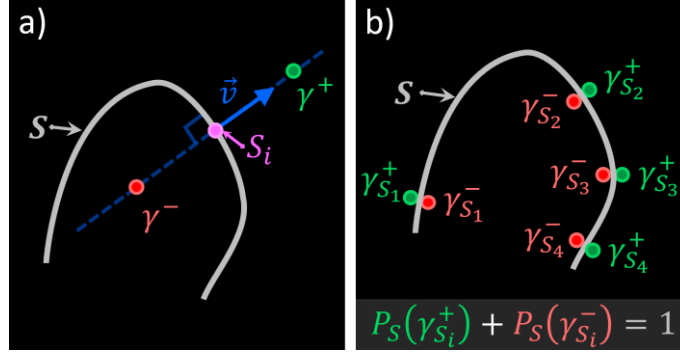


Figure 4-8 : Complementarity of the enclosure probability across the path S . (a) 2 points at an opposed side of S . (b) Multiple complementary points, at opposed sides of S , but with an infinitesimal distance.

Another important characteristic of the probabilities is that P_S should have a value of 1 everywhere inside a closed contour, and a value 0 everywhere outside it. A closed contour can be viewed as any partial contour where γ_i and γ_f are coincident, meaning that all the circles $S(\beta \neq \pi)$ have a null radius. Hence, the equation (23) forces any point outside the shape to have a value of $P_S = 0$, since $P_S(\gamma_\infty) = 0$ and since there are no circular paths $S(\beta)$ to change its value when γ approaches the closed contour. Therefore, P_S is constant both inside and outside the closed contour and varies only at its boundaries. Hence, using equation (27) with $P_S(\gamma^-) = 0$ allows to demonstrate that $P_S(\gamma^+) = 1$.

Finally, if we suppose that S is the partial contour formed by multiple sub-partial contours s_i , then we need that the probability P_S computed on the partial contour S to be the same as the combined probabilities P_{s_i} computed on each sub-partial contour. To make the problem easily solvable, we need to consider that the probability P_S be the sum or subtraction of all P_{s_i} , as described by equation (28). The operator “ $\pm^?$ ” means that the sign is chosen as positive or negative such that P_S respects the previously stated conditions, and will be discussed in section “4.3.2.1 Repulsion optimization”.

$$P_S = \sum_{i=1}^n \pm^? P_{s_i}, \text{ iff } S = \bigcup_{i=1}^n s_i \quad (28)$$

In summary, there are 5 fundamental properties presented in Table 4.1 that must be respected for the probabilities to be consistent with the mathematics and the boundary conditions.

Table 4.1 : List of fundamental properties for the consistency of the probabilities

#	Properties	Description
1	Laws of probability	Each probability is bounded by equation (19).
2	Certainty of exclusion	Any point γ_∞ at an infinite distance of $\gamma_{i,f}$ has a value of $P_S = 0$ (equation (23)).
3	Complementarity	P_S must be complementary on 2 points at each side of a partial contour, when the distance between those points is infinitesimal (equation (27)).
4	Combination of probabilities	P_S is the sum or subtraction of the probabilities given by each sub-partial contour (equation (28)), such that conditions 1 and 2 are respected.
5	Certainty of inclusion	P_S must be 1 inside a closed partial contour, and 0 outside it. Proven with properties #1,2,3.

4.3 Computing the probabilities in an image using EM

Although we explored the theoretical possibility of computing the probabilities of inclusion, this section is required to present how the EM potentials of dipoles allow generating all those probabilities using mathematical convolutions in an image. First, it demonstrates that the equipotential lines are circular when the bi-dimensional dipoles are perpendicular to the partial contour and that they are related to the paths S_C . Then, it shows how multiple potentials can be combined to form a space of probability of belonging to any partial contour P_S , for any pixel in an image composed of multiple non-trivial partial contours.

4.3.1 Circular paths transform using EM potential

This subsection demonstrates that the dipole potential allows to generate the space of all possible circles and to directly determine the value of P_S on a single partial contour S , using a magnetic convolution. Hence, the complexity of analyzing an infinite subset of circles and their intersections with S will be greatly simplified, thanks to its mathematical equivalence with magnetic potentials.

4.3.1.1 EM convolutions

In order to compute EM potentials in an image, it is necessary to use convolutions to reduce computation time and ease the equations, as stated in previous work by Beaini et al. [15,21]. The electric potential P_e of a single charge in any universe of dimension n is given by equation (29), where \mathbf{r} is the Euclidean distance [15]. In a 2D image, the value of n must be 2 to allow for conservation of energy and the use of Gauss theorem. Furthermore, it was shown that the potential

of a dipole can be written as the complex potential given by the partial derivatives in equation (30) [15].

$$P_e = \begin{cases} |\mathbf{r}|^{2-n} & , \quad n \geq 1, \quad n \neq 2 \\ \ln|\mathbf{r}| & , \quad n = 2 \end{cases} \quad (29)$$

$$P_{dip}^\theta \approx \frac{\partial}{\partial x}(P_e) + i \frac{\partial}{\partial y}(P_e) \quad (30)$$

Furthermore, these EM potentials can be easily applied to an intensity image I by using the convolution in equation (13) with the correction factor F (12) [15,21]. In the current paper, I is the matrix with a value of 1 at the thin partial contour and 0 elsewhere, and θ is the direction of the partial contour at any point in the matrix I .

$$F = \max(|\cos(\theta)|, |\sin(\theta)|)^{-1} \Rightarrow 1 \leq F \leq \sqrt{2} \quad (31)$$

$$V_m = (I \circ F \circ e^{i\theta}) * P_{dip}^\theta \quad (32)$$

4.3.1.2 Bi-dimensional EM potential on a line

The first step is to compute the EM potential that is generated by a line between 2 points if the line is composed of a uniform density of dipoles perpendicular to its direction. This is illustrated in Figure 4-9, with the series of dipoles pointing in the \hat{y} direction.

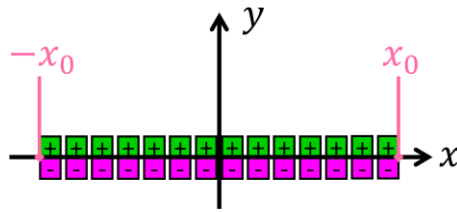


Figure 4-9 : A line on the x axis, composed of dipoles parallel to the y axis.

To compute the potential generated by this line, we first must consider that the potential of a single dipole is the directional derivative of the monopole potential, with the directional derivative in the same direction as the dipole. Then, the contribution of all the dipoles can be taken using a definite integral with the boundaries being the positions $\pm x_0$, shifted by the x position of each point [31,32].

The total potential V_m for the line depicted in Figure 4-9 is then given by equation (33), where P_e is given by equation (29). By choosing $n = 2$, the potential V_m is given by equation (34), with the result given at equation (35), where the values of V_m are bounded by inequality (37) due to the previous arctangent.

$$V_m = \int_{x-x_0}^{x+x_0} \frac{\partial}{\partial y} P_e dx \quad (33)$$

$$V_m = \int_{x-x_0}^{x+x_0} \frac{\partial}{\partial y} \ln(\sqrt{x^2 + y^2}) dx \quad (34)$$

$$V_m = \text{atan}\left(\frac{x+x_0}{y}\right) - \text{atan}\left(\frac{x-x_0}{y}\right) \quad (35)$$

$$-2\pi \leq V_m \leq 2\pi \quad (36)$$

4.3.1.3 Circularity of the equipotential curves

The second step of the sub-section is to prove that the equipotential lines are circular. To prove it, we need to replace the inverse tangent in equation (35) by its complex form with the identity (37) and to define the variable v with the expression (38), which yields to the equation (39). Then, by grouping the x and y together and by using trigonometric identities, we find the equation (40). The complete demonstration is presented in Appendix “D.3.3 Demonstration that equipotential lines are circular”.

$$\text{atan}(x) = \frac{i}{2} \ln\left(\frac{1-ix}{1+ix}\right) \quad (37)$$

$$v \equiv \exp(-2iV_m) \quad (38)$$

$$(v-1)y^2 + (v-1)x^2 + (v+1)2x_0yi - (v-1)x_0^2 = 0 \quad (39)$$

$$x_0^2 \csc^2 V_m = (y + x_0 \cot V_m)^2 + x^2, \quad \{x, y, x_0\} \neq 0 \quad (40)$$

Inspection of equation (40) shows that the equipotential lines are all circular, since each value of V_m gives the equation of a circle. Furthermore, it is the same equation as the one for the circular

path between 2 points given at (20), but with V_m instead of β , which leads to equation (41), since the values are bounded by $\beta = [0, 2\pi]$ and $V_m = [-2\pi, 2\pi]$.

$$|V_m| = \beta \quad (41)$$

4.3.1.4 Circular paths transform

The 3rd step is to be able to compute such potential on a partial contour of any shape. The result of equation (41) means that, for a line L , the magnetic potential V_m at any point γ is equal to the starting angle β of the circle that links the points $\gamma_{i,f}$ (both end of L) to the point γ . Hence, the computation of the probabilities P_S at equation (18) becomes a simple computation of magnetic potential given by equation (42). Furthermore, it is possible to compute all the characteristics of equations (138), (139), (140) and (141) using V_m instead of β and x_0 as the half distance between γ_i and γ_f .

$$P_S(\gamma \subset R) = \frac{|V_m|}{2\pi} \quad (42)$$

The equation (42) is not useful if it can only be applied for a line. Hence, we need the equations (12) and (13) to compute the potential V_m for any thin partial contour in an image, since they allow the superposition of 2 perpendicular dipoles to create a dipole in any direction.

Using equations (30), (12) and (13), we can compute the circular equipotential lines for any partial contour S . The reason why the equipotential lines stay circular is unknown, and a mathematical proof is beyond the scope of this paper. Nevertheless, it is observed numerically with many different shapes in Figure 4-10, where we can see that the expected circular equipotential (in white) match closely the magnetic equipotential lines (in green and pink). There are some numerical errors due mainly to a small error in the angle θ , since the orientation of the partial contours is estimated numerically. I call these equations “circular paths transform”, since it allows to transform a 1D partial contour into a 2D space of circular paths, with each circle passing through both ends of the partial contour and its potential value corresponding to the starting angle β of the circle. However, this allows an alternative way to compute the circular potential is given in the appendix “D.3.2Convolution alternative”.

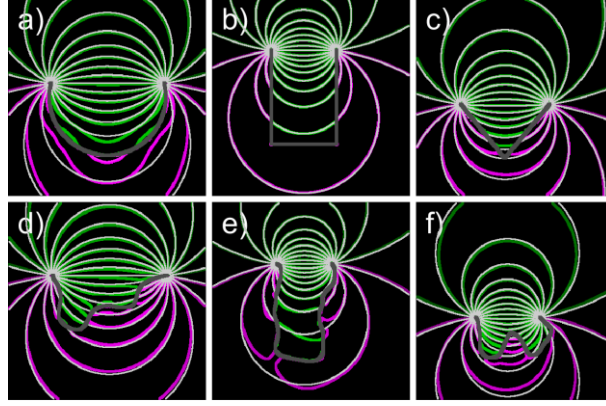


Figure 4-10 : Example of equipotential lines of V_m (green and pink) computed on 6 different partial contours (dark grey), with the light grey lines being the perfectly circular equipotential lines of equations (40) and (41)

4.3.2 Scalar probability superposition

Computing the probability for a partial contour can be useful, but it is usually required to compute the probabilities generated by multiple partial contours in an image. Since the developed method relies on finding all the paths between the extremities of the partial contour, then adding multiple partial contours will require considering the paths between all those extremities. However, such a problem becomes exponentially more complex with each new partial contour that is added and yields to intersecting paths. This section explains how an understanding of magnetic potentials allows to simplify the computation and improve the results through repulsion optimization, double boundary detection and image splitting.

4.3.2.1 Repulsion optimization

There is one major problem when summing different potentials V_m^i , since the dipoles are aligned perpendicularly to the sub-partial contours s_i . This means that the angle θ in equations (12) and (13) can be shifted by 180° , which will shift the sign of V_m^i as seen in equation (43). Hence, there are 2 possible configurations for each sub-partial contour in an image. This problem was raised previously with equation (28), where the sign “ \pm ” was used to mention that it is either an addition or a subtraction, but without certainty.

$$\theta \rightarrow (\theta + \pi) \Rightarrow V_m^i \rightarrow -V_m^i \quad (43)$$

If there is a total of n sub-partial contours, then there should be a total of 2^n solutions, but the absolute value in equation (42) makes half the solutions redundant, meaning that there is a total of 2^{n-1} different solutions. However, there is only one solution that is consistent with equation (28), and it is the one where all the sides of s_i are aligned according to their positive or negative sides. Hence, the magnetic repulsion must be maximized to be consistent with equation (28).

When the repulsion is maximized, there will be multiple regions that form a constant potential as discussed in a previous paper by Beaini et al. [15]. At the boundary condition, a closed shape with all the dipoles aligned will generate 2 regions of constant potential with no gradient \mathbf{E} except at the boundaries where \mathbf{E} is high. In case the dipoles are not aligned, the value of \mathbf{E} will vary smoothly between its minimum and maximum. Therefore, the distribution of \mathbf{E} will be more split when the repulsion is maximized. Hence, we define the maximization parameter to be the variance Ω of $|\mathbf{E}|^2$ depicted in equation (44), meaning that Ω must be maximized to maximize the repulsion.

$$\Omega = \text{Var}(|\mathbf{E}|^2) \quad (44)$$

Since there are 2^{n-1} configurations, then it is preferable to use an optimization algorithm when n is large to avoid long computing time. An algorithm that was developed and tested consist of creating a list G which contains each individual index i , plus multiple groups of indices that are chosen according to their magnetic interaction. For example, the sub-partial contours s_i that connect with each other with a potential of $V_m > V_{th1}$ will form a group, those with a potential $V_m > V_{th2}$ will form another group.

Then, the potential $V_m^{G_k}$ of each element of G are flipped and tested to see their impact on Ω . If Ω is increased, then the elements of G_k are permanently flipped. This algorithm is described in Figure 4-11 and was observed to work in most cases. If the number of elements are high, then the algorithm might end up in a local maximum. To avoid such problems, it can be used on different randomized initial orientations. Once each of them is optimized through the algorithm, the best solution must be chosen as the one with the lowest value of Ω .

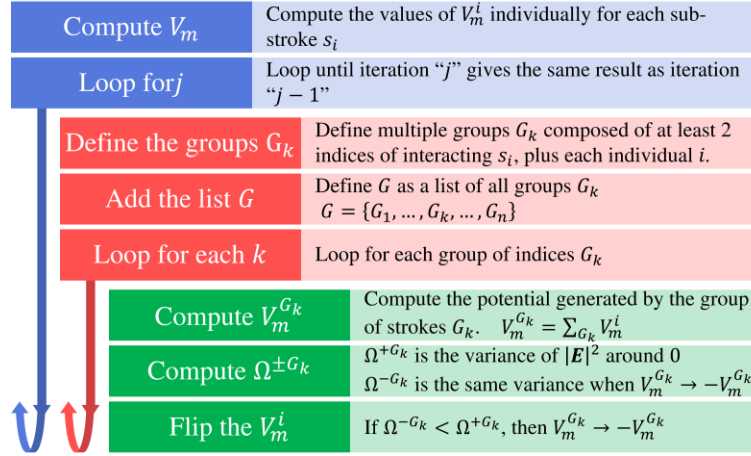


Figure 4-11 : Algorithm used for repulsion optimization by flipping the magnetic orientation of each individual or group of sub-partial contours G_k .

An example of such optimization is observed in Figure 4-12, where the partial contours are extracted via the canny algorithm [63] with a high threshold. We can see that after the repulsion optimization, the high potentials $|V_m|$ are concentrated in the regions there are shapes, and the near zero potentials are between those shapes. It is to note that there are small regions where $|V_m| > 2\pi$, which are saturated in the Figure 4-12 and Figure 4-13.

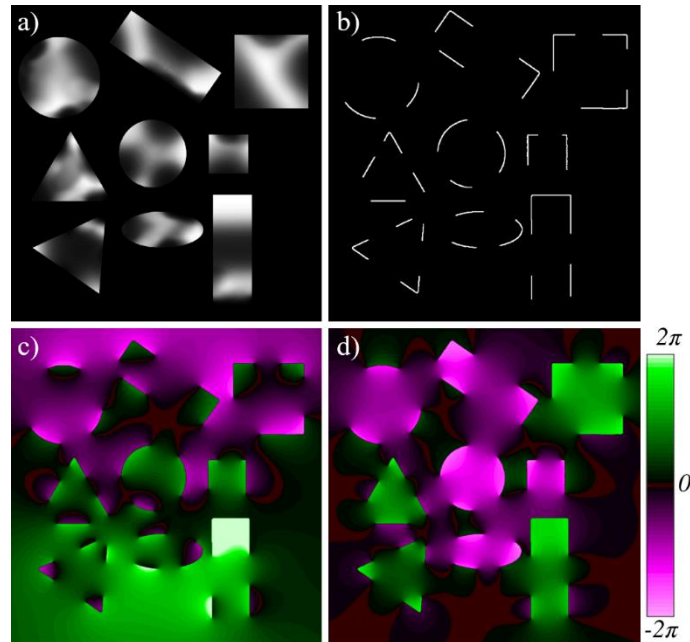


Figure 4-12 : (a) Artificial image composed of different nearby shapes (b) Extracted partial contours using Canny [63]; (c) Resulting V_m in the initial orientation; (d) resulting V_m after the repulsion optimization.

The algorithm in Figure 4-11 was tested with the 28 partial contours s_i of Figure 4-12, and the result was compared to the brute force optimization that minimized Ω by testing the 2^{27} different configurations. The results were the same, but the computation time was around a 10^6 times faster using the algorithm. This test was done with different images, including Figure 4-13, and the results were always the same, which shows that the algorithm converges to an optimal result.

4.3.2.2 Double boundaries

In some cases, a partial contour will be at the boundaries of 2 different regions, which means that its contribution should be doubled to consider both regions. There are 2 equivalent ways of doing it, which are either to double the value of F in equation (13) or to create a second partial contour adjacent to the first one. An example with a few adjacent shapes is presented at Figure 4-13, where we can see the improvement of the potential when the double boundary is considered. One important improvement is the reduced potential between the shapes, so the high potential is mainly concentrated within the shapes. Another one is that the double boundary produces 2 clearer sides when it is considered, as seen by the circle and the triangle at the left. Furthermore, the 2 regions of the top rectangle are only distinguishable when the double boundaries are considered.

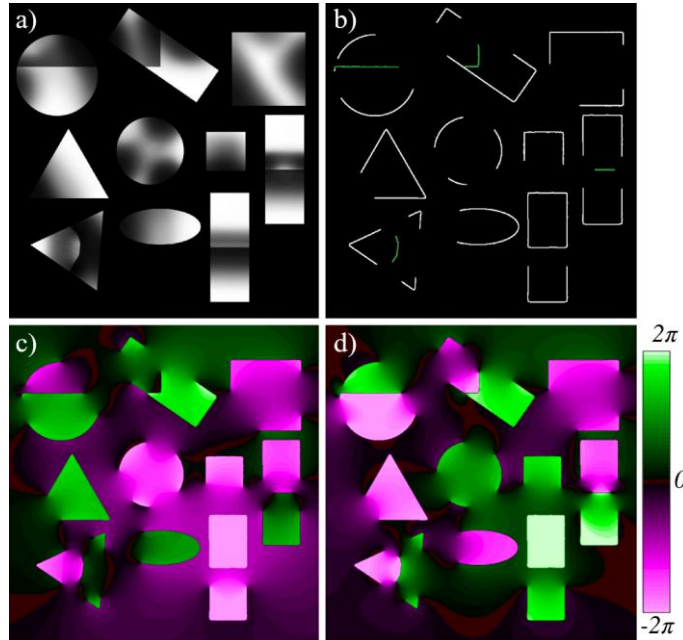


Figure 4-13 : (a) Artificial image composed of different adjacent shapes (b) Extracted partial contours using Canny [63], with the double boundaries in green; (c) Resulting V_m without the double boundary; (d) resulting V_m with the double boundary.

4.3.2.3 Image splitting by attraction elimination

In the case where many different shapes are present in a single image, the repulsion optimization will still yield in some adjacent shapes that produce an attractive field between each other, since one will have a positive V_m , while the other will have a negative V_m . Since those shapes will be sure to not belong together, then they can be split into 2 new images that do not interact together. The algorithm to decide how to split them is presented at Figure 4-14, with the goal of reducing the initial potential image into multiple as much sub-images as possible, without loss of information. It is to note that this step is not mandatory since it increases the total computation time, although it usually improves the results. Also, some partial contours might be in different sub-images, since they can belong to different groups. An example of the algorithm is presented in Figure 4-15.

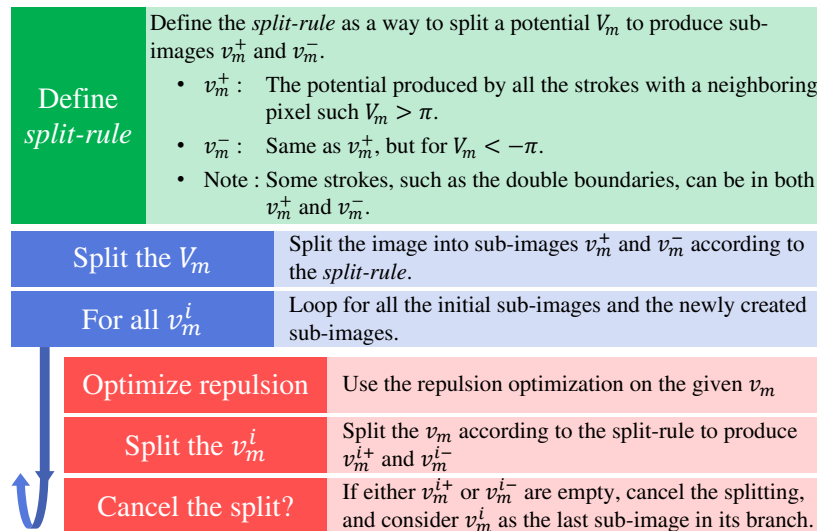


Figure 4-14 : The algorithm used for the image splitting into multiple sub-images

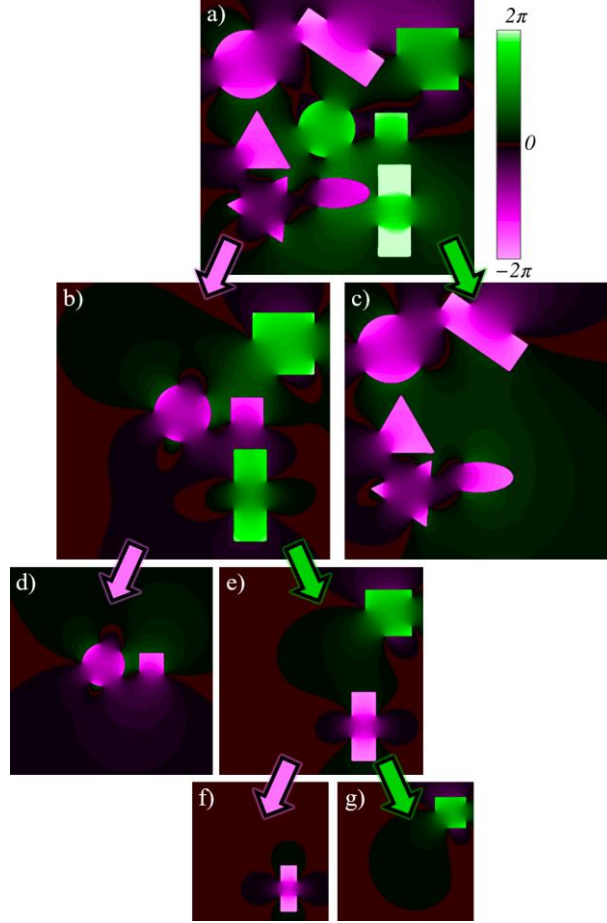


Figure 4-15 : Example of image splitting process; (a, b, e) The temporary states of the splitting;
(c, d, f, g) The final set of split potentials v_m^l

4.4 Important properties

With the knowledge of the previous sections, we know how to properly compute the probability P_S using EM convolutions, but we did not discuss the interesting properties that arise. Hence, this section will cover some special features such as the weight adjustments, the equipotential line destination, the possibility of generalizing it in 3D and the information estimation.

4.4.1 About the probabilities

This subsection will focus on covering the closed shapes, the invalid probabilities and the possibility of adjusting the weight of each probability.

4.4.1.1 Closed shapes

In some cases, a partial contour may be already closed, which means that P_S^+ must be 1 inside S , and P_S^- must be 0 outside S , as stated in the fundamental properties at Table 4.1. To prove it, we first use Gauss theorem, since it was demonstrated by Beaini et al. [15] that V_m is constant both inside and outside of S [31,32]. Then, we know that the potential is null at a point γ_∞ infinitely far, as seen in equation (35). Finally, we know from equation (143) that crossing the partial contour leads to a potential variation $\Delta V_m = \pm 2\pi$, which means the value is 0 outside the partial contour and $\pm 2\pi$ inside it. Hence, knowing from equation (42) that $P_S = |V_m|/2\pi$, we demonstrate that P_S is 0 outside the shape and 1 inside it, which is consistent with the properties at Table 4.1.

Furthermore, we know from Beaini et al. [15] that Gauss theorem will only give a constant potential inside a shape if and only if the potential of a charge P_e is proportional to the equation (29) [31,32]. We also know that the probabilities are only consistent if we use dipoles that are perpendicular to the contour. Hence, we conclude that the potential $V_m = (I \circ F \circ e^{j\theta}) * P_{\text{dip}}^\theta$ given at equation (13) is the only possible potential that can be used for the computation of the probability of inclusion inside a partial contour S , with $P_S = |V_m|/2\pi$ (42).

In summary, the developed method is believed to be the only possible way to compute P_S using potential convolutions. This is because a valid P_S requires a probability of 1 inside a closed partial contour and 0 outside it, which requires conservation of energy and is only possible via EM kernels.

4.4.1.2 Invalid probabilities

In some other cases, the probabilities computed using equation (42) will be greater than 1, which is invalid mathematically. Most of those times, the probability will be in the interval $[1, 1.10]$, which is simply a numerical error. Most of those errors are one-off occurrences and can be solved by a median filter, while the rest can simply be rounded to the value 1. However, other cases will have a value that is in the interval $[1.10, 2]$, which happens when the given point is inside 2 shapes simultaneously. This is the result of an attraction instead of repulsion or of a self-containing shape. Most of those problems are solved or reduced via the image splitting described in section “4.3.2.3 Image splitting by attraction elimination”. However, the only way to permanently solve this problem is to saturate the values of P_S for a maximum of 1, which is the approach used in this paper.

4.4.1.3 Weight adjustments

The proposed method allows computing the probability P_S using equation (42), but only if an equal weight is attributed to each of the circular equipotential. As it was discussed in section “4.4.1.1 Closed shapes”, it is impossible to change the potential to add more weight for the shortest equipotential. However, it is possible to weight the probability P_S by using a smooth-step function to obtain a weighted probability W_S in equation (45), which is based on the Hermite polynomials and is valid for any value of $P_S = [0, 1]$ [111]. An example of the smooth-step function for $K = 2$ is given in equation (46).

A weight function will only work if it is bounded by $[0, 1]$, strictly increasing and antisymmetric around $P_S = 0.5$. Since the smooth-step function respects those conditions, then it respects all the properties required for the probabilities to stay consistent with the fundamental properties given in Table 4.1.

In the case described in section “4.3.2.3 Image splitting by attraction elimination”, it was explained that the probabilities will be better if the potential image is split into multiple sub-images. In that case, the total weighted probability W_S is considered as the maximum value of all the weights of the sub-images w_S^i , as described in equation (47). Although equation (47) is not consistent with Table 4.1, it allows to determine what is the maximum probability of belonging inside a shape, which is still a relevant information. Otherwise, we can still access all the w_S^i individually.

$$W_S = P_S^{K+1} \sum_{k=0}^K \binom{K+k}{k} \binom{2K+1}{K-k} (-P_S)^k \quad (45)$$

$$W_{S(K=2)} = 6P_S^5 - 15P_S^4 + 10P_S^3 \quad (46)$$

$$W_S = \max(w_S^i) \quad (47)$$

4.4.2 Additional features

This subsection will cover other additional features that can be obtained by the P_S or V_m , but without discussing them thoroughly. Those features include the equipotential line destinations, the possibility of computing uncertain partial contours and the possibility of analyzing 3D shapes.

4.4.2.1 Equipotential lines destinations

An interesting fact to note about the equipotential lines is that they always seem to pass through the extremities of the partial contours, even when the image is complex, such as Figure 4-12 and Figure 4-13. In those images, we can see that only a few equipotential lines avoid the extremities, and it happens near the corners where the numerical error is higher. Also, some equipotential lines will cross the partial contours and will be subject to the transformation at equation (143), but they will eventually reach the extremities. This fact means that, by using a variable threshold value on the potential, it is possible to obtain different hypothetical shapes that are formed by the given partial contours.

4.4.2.2 Uncertain partial contour

In some cases, a part of a partial contour might not be certain to be an actual contour, and setting its partial contour value to either 0 or 1 according to equation (13) might not be the best option. In that case, the matrix I which is usually composed of 0 and 1, can be changed to be any real value bounded by 0 and 1. This will be equivalent of reducing the weight associated to the specific partial contour. For example, the P_S of a closed partial contour with a value of 0.7 will be 0.7 inside it, and 0 outside it.

4.4.2.3 Probability analysis for 3D shapes

The work from the current paper can also be generalized for a 3D partial surface S_3 , where the proposed method would be able to compute the probability of belonging inside the solid. To do so, we need to use the equation (30) with a value of $n = 3$ and replace the factor 2π in equation (42) by the factor 4π . Furthermore, the equations (12) and (13) need to be changed to consider 2 angles θ and ϕ to take into account the 3D orientation, such that each voxel in I will have an orientation perpendicular to the surface at this point.

Using the equations in 3D will not produce circular shapes anymore, but complex 3D shapes. However, this does not impact the ability to compute the probabilities, since the results will still be consistent with the properties of Table 4.1 if the word “partial contour” is replaced by “surface”. Furthermore, in the boundary condition where the surface $S_3(x, y, z)$ is independent of z , then the computed probability P_{S_3} in any xy -plane will produce circular equipotential lines, and P_{S_3} will be the same as the probability P_S computed with $n = 2$ on a partial contour S , where $S = S_3$.

4.4.3 3D information estimation from 1D partial contours

Another aspect of the proposed approach is that it allows estimating the original image based only on the information available with the partial partial contours, which is impressive since the partial contours are 1D information, while an image is a 3D information.

In fact, the image I composed of the partial contours S represent 1D information, since the partial contours are thin, and their value is either 0 or 1. However, the computation of W_S using equation (46) generates 3D information, since it fills all the pixels in the image with a value in the range $[0, 1]$. Hence, a surprising characteristic of CAMERA-I-PIPE is that the probabilistic reconstruction allows estimating the original 3D image (height, width and intensity) using only the 1D partial contours, as seen in Figure 4-16.

Although it is impossible to obtain the same image since most information is lost by taking the partial contours, the estimated results are extremely similar both in shape and in intensity to the original image. Hence, Figure 4-16 shows that the probability computation is consistent with the expectations that W_S allows estimating the original image using only its partial contours.

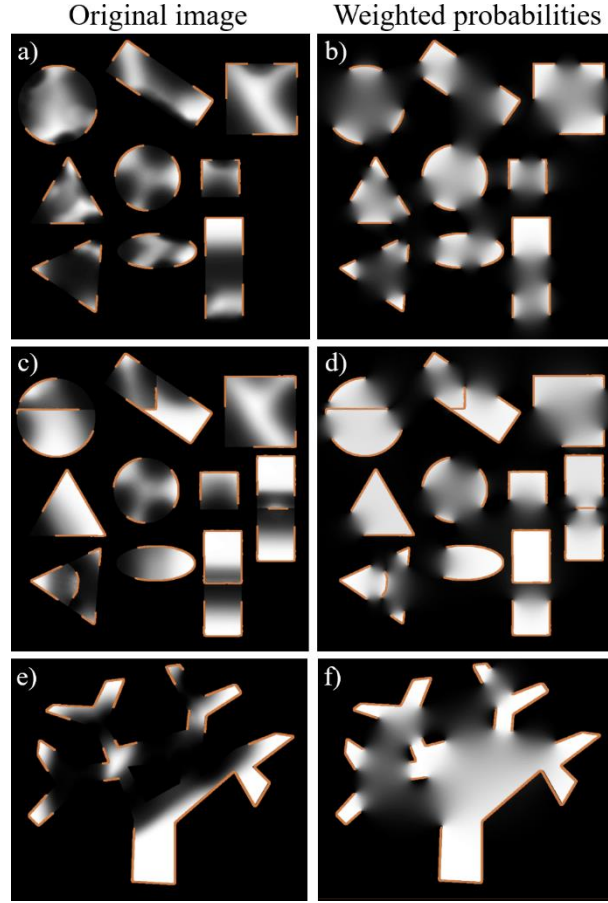


Figure 4-16 : Comparison of the original images with the probabilistic reconstruction. (a, c, e) Original synthetic image composed of different shapes with the partial contours (orange); (b, d, f) Probabilistic weighted reconstruction based on the partial contours (orange).

4.5 Conclusion

The work presented in this paper detailed the development of the CAMERA-I-PIIPE method, which allows computing a spatial probability of inclusion P_S according to initial partial contours. To do so, it explained how we can use an uncountable set of subset paths to P_S , called and how to generate such a set using all the possible circular paths via a simple potential convolution. Then, it showed how the magnetized contours can be manipulated to compute P_S on complex images with multiple contours. Finally, different features were studied, such as the double boundaries, the weight adjustment technique, the uncertain edges and the information estimation.

This paper is a precursor to numerous possible studies for computer vision applications since it created a novel approach that generates a space of probabilities based only on partial contours. For

the first time, it is possible to directly combine contour information and region information for image processing. A continuation of this work could focus on developing specific applications in different computer vision fields such as saliency, image segmentation and contour completion. For now, most methods for these applications consider either the region information or the edge information. Hence, we expect that they will benefit from the promising results of the current work since it should allow combining edge-based and region-based approaches together.

Acknowledgment

We would like to thank NSERC, through the discovery grant program RGPIN-2014-06289, and FRQNT/INTER for their financial support as well as MEDITIS (Biomedical technologies training program) through NSERC (FONCER) initiative.

CHAPTER 5 ARTICLE 3: FAST AND OPTIMAL LAPLACIAN SOLVER FOR GRADIENT-DOMAIN IMAGE EDITING USING GREEN FUNCTION CONVOLUTION

Dominique Beaini^a, Sofiane Achiche^a, Fabrice Nonez^b, Olivier Brochu Dufour^a, Cédric Leblond-Ménard^a, Mahdis Asaadi^a, Maxime Raison^a

^aPolytechnique Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

^bUniversity of Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

Submitted to Advances in Engineering Software, Elsevier, February 2019

Abstract

In computer vision, the gradient and Laplacian of an image are used in different applications, such as edge detection, feature extraction, and seamless image cloning. Computing the gradient of an image is straightforward since numerical derivatives are available in most computer vision toolboxes. However, the reverse problem is more difficult, since computing an image from its gradient requires to solve the Laplacian equation (also called Poisson equation). Current discrete methods are either slow or require heavy parallel computing. The objective of this paper is to present a novel fast and robust method of solving the image gradient or Laplacian with minimal error, which can be used for gradient-domain editing. By using a single convolution based on a numerical Green's function, the whole process is faster and straightforward to implement with different computer vision libraries. It can also be optimized on a GPU using fast Fourier transforms and can easily be generalized for an n-dimension image. The tests show that, for images of resolution 801x1200, the proposed GFC can solve 100 Laplacian in parallel in around 1.0 milliseconds (ms). This is orders of magnitude faster than our nearest competitor which requires 294ms for a single image. Furthermore, we prove mathematically and demonstrate empirically that the proposed method is the least-error solver for gradient domain editing. The developed method is also validated with examples of Poisson blending, gradient removal, and the proposed gradient domain merging (GDM). Finally, we present how the GDM can be leveraged in future works for convolutional neural networks (CNN).

Keywords: Computer vision, Poisson image editing, seamless cloning, Green function convolution, Gradient Laplacian Solver, gradient domain editing

5.1 Introduction

In computer vision and signal processing, the images can be interpreted as numerical potentials, especially when there is an interest in their gradient (\mathbf{E}). For example, early computer vision system algorithms relied a lot on numerical gradients and Laplacian [15,16,21,57,59,61] to extract important information about edges and image boundaries. They are computed using simple convolution kernel such as Sobel [34–36]. More recently, there have been growing interest in gradient domain image editing (GDIE) applications, which aim at editing or creating images from its gradient [57,59–61,112].

Although computing the gradient or Laplacian of an image is straightforward, the reverse problem of computing the image from its gradient is a non-trivial task. In fact, this problem requires to solve a differential equation [57,59,61] without knowing if an exact solution exists. When the gradient is computed from an image, it always generates a conservative field, meaning that the field can be integrated to obtain a potential (the original image). For gradient domain image editing (GDIE), a non-conservative perturbation is voluntarily introduced to the gradient, meaning that the resulting field cannot be integrated into an exact solution.

Nevertheless, it is still interesting to solve the non-conservative gradient since it leads to many gradient-domain editing applications, such as gradient erasing, seamless cloning and vectorization with diffusion curves [57,59–61,112]. Furthermore, Bhat et al. presented a whole framework of gradient-domain image editing with unique and useful applications such as color filtering and edge sharpening [58].

The first method to solve the image Laplacian (also called Poisson’s equation) was presented in 2003 by Perez et al. [57], which proposed to solve the differential equation by iteratively minimizing the variational problem. Other research followed by optimizing the computation speed and error [57,60], others used the Jacobi method [59], and McCann proposes a multi-grid solver [112]. These approaches converge to the approximate solution, but they are harder to implement since they are iterative, which also makes them slower to compute. An alternate way of solving the Poisson equation is proposed by Tanaka [61] by modifying the Poisson problem into a closed-form problem using cosine transforms.

More recent methods solve this problem by using methods based on multipoles and Green's function [113–115]. However, they are only implemented for diffusion curves in vector graphics, and not for gradient domain image editing. Furthermore, the Green's function methods [113–115] and the Tanaka method [61] propose analytical solutions for continuous space, but we propose a numerical solution developed for a discrete space. Hence, our method is more suitable for discrete images and is demonstrated to have a lower error than Tanaka in section 5.3.1.4.

The objective of this paper is to present a novel fast and robust method of solving the image gradient or Laplacian with minimal error, which can be used for gradient-domain editing.

In the research work presented here, a novel method is proposed called Green Function Convolution (GFC), which allows solving any modified gradient. In the case of a non-conservative field, the proposed GFC method is proven to find the best possible approximation in terms of gradient error. In fact, we mathematically prove in section 5.2.2 and empirically in section 5.3.1.4 that GFC is the optimal possible solver for any perturbation added to the gradient, meaning that gradient domain editing can be done with minimal error.

Our contributions are summarized below:

Simple, fast and optimal gradient/Laplacian solver. The implementation that we propose is simple, requiring only a few lines of code using any library that implements the 2D fast Fourier transform (FFT). The implementation is also significantly faster than competing methods since we showed in Figure 5-2 a 170x improvement compared to Tanaka's method [61], thanks to our graphics processing units (GPU) implementation. We also showed that our GFC solver can process 100 images in 1ms using Pytorch, making it the fastest method available for discrete images. Finally, we demonstrated mathematically in section 5.2.2 and empirically in section 5.3.1.4 that GFC is optimal in the sense that it is the least-error solver for gradient domain editing.

Gradient domain merging applications. Inspired by edge saliency sharpening techniques [58] and recent edge detection methods [25,64], we develop a novel method of reducing texture information and enhancing boundaries contrast. Our work proposes the first method to use machine learning edge detectors for this purpose. With our GFC solver that relies mainly on FFT, we show that the solver can be implemented in deep learning libraries such as Tensorflow and Pytorch and can be leveraged in future works for machine learning applications.

5.2 Computing the image from its gradient or Laplacian

To understand how to compute the image from its gradient field or Laplacian, we first focus on the mathematical understanding of the Green's function and its ability to solve any Laplacian [33]. We will show how to find the appropriate Green's function and how to solve either the gradient or the Laplacian. Then, we will demonstrate mathematically that using Green's function is the optimal tool when there is a non-conservative perturbation that is added to the gradient field.

5.2.1 Green's function to solve the Laplacian

This subsection explains how the Green's function can be used to theoretically solve a Laplacian (Poisson equation) on any signal.

First, we define the gradient field \mathbf{E} of a signal (image) I in equation (48), where ∇ is the n^{th} dimension gradient operator. In many applications such as computer vision, computing the gradient is very simple to do using numerical derivatives such as the Sobel method [34,63]. However, the reverse problem of finding the signal (or image) I from the field \mathbf{E} defined in (49) is not trivial, since the curvilinear integral \int_C is not always defined. In fact, the integral (49) is only defined in the case of a conservative field. In the case of gradient domain image editing (GDIE), the field is modified via a non-conservative perturbation, which renders equation (49) unsolvable.

$$\mathbf{E} = \nabla I \quad (48)$$

$$V_E = - \int_C \mathbf{E} \cdot d\mathbf{l} \quad (49)$$

Instead of solving the gradient, most approaches focus on solving the Laplacian (also known as Poisson equation) defined in (50).

$$L = \nabla \cdot \mathbf{E} = \nabla^2 I \quad (50)$$

Since the Laplacian is a differential equation, we propose to solve it using a Green's function, which is defined as a function that solves a differential equation via convolution [33]. This definition is expressed in (51), where V_{mono} is the Green's function of the Laplacian ∇^2 , $\nabla \cdot$ is the divergence operator and $*$ is the convolution operator. The notation V_{mono} is chosen since it is

based on our previous work concerning electromagnetic potentials in images [15,16], where the potentials are in fact the 2D Green's function [33]. The equation (51) is at the heart of our proposed Green function convolution (GFC) method.

$$\begin{aligned} I &= (\nabla^2 I) * V_{\text{mono}} \\ I &= (\nabla \cdot \mathbf{E}) * V_{\text{mono}} \end{aligned} \quad (51)$$

Other GDIE methods proposed using multipoles and Green's function based solvers [113,115]. However, we differentiate ourselves from their work [113,115] by focusing on a purely numerical solution, instead of solving the Green's function analytically.

The Green's function V_{mono} is given in equation (52), with the constant S_{n-1} given in equation (53) where Γ is the gamma function and r is the Euclidean distance [15,33,113,115]. For the other methods based on the Green's function [113–115], V_{mono} is modified to account for the rectangular boundary around the image, which is not required for us since we compute V_{mono} numerically.

$$\begin{aligned} V_{\text{mono}} &= \frac{-1}{S_{n-1}} \int r^{(1-n)} dr \\ V_{\text{mono}} &= \frac{-1}{S_{n-1}} \cdot \begin{cases} \ln(r), & n = 2 \\ \frac{r^{2-n}}{n-2}, & n \neq 2 \end{cases}, \quad n \in \mathbb{N}^* \end{aligned} \quad (52)$$

$$S_{n-1} = \frac{2\pi^{n/2}}{\Gamma(n/2)}, \quad S_{n=2} = 2\pi \quad (53)$$

In our previous work [15,16], we used a physics-inspired method, which convolved electromagnetic dipoles in the direction of the gradient for partial contour analysis. Those dipole potentials V_{dip} are in fact the Green's function of the gradient \mathbf{E} , meaning that they directly solve the gradient without first computing the Laplacian. The gradient solver using V_{dip} is presented in equation (54), where n is the number of dimensions ($n = 2$ for an image) and x_i is the axis of each dimension. Hence, each dipole is convolved with each component of the gradient. Notice that the definition consists of moving the divergence operator $\nabla \cdot$ from \mathbf{E} to V_{mono} in equation (51).

$$I = \sum_{i=1}^n E_{x_i} * \underbrace{\frac{\partial}{\partial x_i}(V_{\text{mono}})}_{\equiv V_{dip}^i} \quad (54)$$

Although definitions (51) and (54) are both valid, the current paper focuses on the definition given by (51) since it requires a single convolution. Furthermore, V_{dip} was developed in previous work to account for the strong electromagnetic inspiration. However, since it is no longer important in our current work, we will favor solving the gradient and Laplacian using equation (51).

5.2.2 Proof of optimal result for any perturbations in the gradient

The above-presented mathematical equations (51) and (54) showed how to re-compute the image I from its gradient \mathbf{E} or Laplacian L using the convolutions with the Green's function V_{mono} . However, there are many GDIE applications that require adding a voluntary non-conservative perturbation to the field, such as those presented in section 5.3.2. The perturbed field is noted \mathbf{E}_p , while the computed field and potentials are respectively \mathbf{E}_c and I_c .

Since the perturbation can be non-conservative, the field \mathbf{E}_p does not have an associated potential and cannot be solved exactly. Hence, there is a need to find the conservative field \mathbf{E}_c that is the best possible approximation of \mathbf{E}_p . This section will prove that equations (51) and (54) give the optimal I_c and \mathbf{E}_c for any possible perturbation. Thus, it proves that the proposed GFC method is robust to perturbation and that it will converge to the least error solution, where the error is defined as $\epsilon = |\mathbf{E}_p - \mathbf{E}_c|$.

First, using Hilbert projection theorem, we know that the minimum-error solution is given when ϵ is orthogonal to any conservative field ∇U at any point [116]. Hence, we need to prove that $F = 0$ (equation (55)), where $d\mu$ is the infinitesimal hyper-volume for the integration.

$$F = \int_{\mathbb{R}^n} \left[\underbrace{[\mathbf{E}_p - \mathbf{E}_c]}_{\equiv \epsilon} \cdot \nabla U \right] d\mu = 0 \quad (55)$$

To prove (55), we first replace the value of \mathbf{E}_c by its correspondence \mathbf{E}_p , as given in equation (56). Then, we substitute I_c by $((\nabla \cdot \mathbf{E}_p) * V_{\text{mono}})$ according to equation (51). We also define the variable A as a temporary variable to make it easier to follow the proof.

$$\begin{aligned}
F &= \int_{\mathbb{R}^n} \left[[\mathbf{E}_p - \nabla(I_c)] \cdot \nabla U \right] d\mu \\
F &= \int_{\mathbb{R}^n} \left[\underbrace{\left[\mathbf{E}_p - \nabla \left((\nabla \cdot \mathbf{E}_p) * V_{\text{mono}} \right) \right]}_{\equiv A} \cdot \nabla U \right] d\mu \\
F &= \int_{\mathbb{R}^n} [A \cdot \nabla U] d\mu
\end{aligned} \tag{56}$$

By adding and subtracting the term $(\nabla \cdot A)U$ inside the integral, we obtain equation (57). Then, we use the divergence properties in equation (58) to regroup the positive terms inside an integral and the negative terms in another.

$$F = \int_{\mathbb{R}^n} [(A \cdot \nabla U) + (\nabla \cdot A)U - (\nabla \cdot A)U] d\mu \tag{57}$$

$$F = \underbrace{\int_{\mathbb{R}^n} [\nabla \cdot (AU)] d\mu}_{\equiv B} - \int_{\mathbb{R}^n} (\nabla \cdot A)U d\mu \tag{58}$$

In equation (58), the term noted B has a value of 0 and is canceled. This is due to Gauss's theorem which states that the integral of a divergence is the integral of the flux outside the surface [31,33]. However, as it is explained later in section 5.2.3.2, since a zero padding is added around the image, then the flux is 0 at every point of the boundaries of the surface. Therefore, equation (59) is the remaining term of equation (58), where the value of A is substituted by its definition in equation (55).

$$F = - \int_{\mathbb{R}^n} \left[\nabla \cdot \left[\mathbf{E}_p - \nabla \left((\nabla \cdot \mathbf{E}_p) * V_{\text{mono}} \right) \right] \right] U d\mu \tag{59}$$

Then, equation (60) distributes the derivative operators according to the properties of the sum and the convolutions.

$$\begin{aligned}
F &= - \int_{\mathbb{R}^n} \left[\nabla \cdot \mathbf{E}_p - \nabla^2 \left((\nabla \cdot \mathbf{E}_p) * V_{\text{mono}} \right) \right] U \, d\mu \\
F &= - \int_{\mathbb{R}^n} \left[\nabla \cdot \mathbf{E}_p - \left((\nabla \cdot \mathbf{E}_p) * \nabla^2 V_{\text{mono}} \right) \right] U \, d\mu
\end{aligned} \tag{60}$$

Finally, since V_{mono} is the Green's function of ∇^2 , then by definition $\nabla^2 V_{\text{mono}}$ is a Dirac's delta δ [33]. Knowing that for any function f convoluted with a Dirac's delta δ , we have $f * \delta = f$ [33], equation (61) gives us the final result $F = 0$. Hence, $\mathbf{E}_p - \mathbf{E}_c$ is orthogonal to any other field. According to Hilbert's theorem, the conservative field \mathbf{E}_c has the least error when compared to the perturbed field \mathbf{E}_p .

$$\begin{aligned}
F &= - \int_{\mathbb{R}^n} \left[\nabla \cdot \mathbf{E}_p - \left((\nabla \cdot \mathbf{E}_p) * \delta \right) \right] U \, d\mu \\
F &= - \int_{\mathbb{R}^n} [\nabla \cdot \mathbf{E}_p - \nabla \cdot \mathbf{E}_p] U \, d\mu \\
F &= 0
\end{aligned} \tag{61}$$

This completes the proof that the GFC method allows computing the field \mathbf{E}_c and the potential I_c which are the optimal conservative approximation for any perturbed field \mathbf{E}_p . Hence, the GFC method will always converge to the least-error possible solution, meaning that it is robust to any change or perturbation added to the field. This proof is also valid in the case of an n-dimension image or signal, not just in 2D.

Although we prove that the proposed GFC method is a least-error solver, it does not mean that the cited competing methods are not also least-error solvers. However, section 5.3.1.4 demonstrates empirically that GFC has consistently lower error than the competing Perez [57] and Tanaka [61] methods, thus supporting the proof that the GFC solver is optimal in the case of added perturbation.

5.2.3 Numerical implementation

The mathematical proof of section 5.2.2 demonstrated that the proposed GFC method gives the optimal result without any iterative computation, even when a perturbation is added to the gradient. The current section will show how to implement the optimal GFC solver numerically using fast Fourier transforms (FFT).

5.2.3.1 Problems with the Green's function

Although the n^{th} dimension Green's function is defined in equation (52), it cannot be directly applied to an image. The reason is that the function is defined in a continuous infinite space, while images are a bounded discrete space.

Other works propose to use boundary conditions [61,115] or to find the analytical Green's function for a rectangular space [113]. In our work, we propose using a purely numerical solution, that can also be generalized to non-Laplacian operators.

Advantages of our numerical method are that it is simple to implement, fast to compute and considers the grid structure of the space and the grid nature of the FFT.

5.2.3.2 The numerical Green's function

This subsection shows how to build the numerical Green's function using the convolution theorem and the numerical Fourier transform.

First, the images, gradient, and Laplacian are defined as 2D matrices with an intensity value at each point. For the gradient, there are 2 matrices, one for the horizontal direction and one for the vertical direction. For each pixel in an image, there is an associated Laplacian and gradient.

The numerical gradient and Laplace operators are defined as smaller kernel matrices, which are applied on images via convolution. The numerical Laplace operator is given by equation (62) [34,35].

$$K_{\nabla^2} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (62)$$

We also know that, by definition, the Green's function V_{mono} convoluted by the Laplacian operator K_{∇^2} should give the Dirac's delta δ [33]. This relation is shown in equation (63) where $*$ is the convolution operator.

$$(K_{\nabla^2} * V_{mono}) = \delta \quad (63)$$

We also know that the convolution is defined as the product in the Fourier domain as given in equation (64) [33], where \mathcal{F} is the Fourier transform, \mathcal{F}^{-1} is the inverse Fourier transform, and A, B are any function.

$$A * B = \mathcal{F}^{-1}(\mathcal{F}(A) \circ \mathcal{F}(B)) \quad (64)$$

Numerically, the Fourier transform is fast and easy to compute using Fast Fourier Transform (FFT) algorithms.

Using equation (63) with the convolution theorem (64), we obtain equation (65). Then we isolate V_{mono} in equation (66) to obtain a mathematical definition of the Green's function V_{mono} in the Fourier domain, which we note $V_{mono}^{\mathcal{F}}$.

$$\mathcal{F}^{-1}(\mathcal{F}(K_{\nabla^2}) \circ \mathcal{F}(V_{mono})) = \delta \quad (65)$$

$$V_{mono}^{\mathcal{F}} \equiv \mathcal{F}(V_{mono}) = \frac{\mathcal{F}(\delta)}{\mathcal{F}(K_{\nabla^2})} \quad (66)$$

For this definition to work in a discrete environment, we need the matrices to all be the same size as the image I . Hence, we define the zero-padded matrices \check{K}_{∇^2} and $\check{\delta}$ in equations (67) and (68), where the top left corner are the 3×3 Laplacian and Dirac kernels and the rest of the matrices is 0-valued.

$$\check{K}_{\nabla^2} \equiv \begin{bmatrix} 0 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & & \\ 0 & -1 & 0 & & \\ \vdots & & & \ddots & \\ 0 & & & & 0 \end{bmatrix} \quad (67)$$

$\underbrace{\hspace{10em}}_{size(I)}$

$$\check{\delta} \equiv \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & \\ 0 & 0 & 0 & & \\ \vdots & & & \ddots & \\ 0 & & & & 0 \end{bmatrix} \quad (68)$$

$\underbrace{\hspace{10em}}_{size(I)}$

Using the definitions (67) and (68) alongside equation (66), we find the Green's function in the Fourier domain $\check{V}_{mono}^{\mathcal{F}}$ in equation (69).

$$\check{V}_{mono}^{\mathcal{F}} = \frac{\mathcal{F}(\delta)}{\mathcal{F}(\check{K}_{\nabla^2})} \quad (69)$$

Finally, using the Laplacian solver of equation (51) we can solve the least-error potential I_c from its Laplacian L . The result is given in equation (70), where \mathcal{R} is the real part of a complex number and \circ is the Hadamard element-wise product. We note that $I_c = I$ if the right integration constant c is used.

$$I_c = \mathcal{R} \left(\mathcal{F}^{-1} \left(\mathcal{F}(L) \circ \check{V}_{mono}^{\mathcal{F}} \right) \right) + c \quad (70)$$

In the cases where the boundaries need to be preserved, then it is suggested to add 3-pixel padding to I before passing to the gradient domain, then retrieve the constant c such as the padded region in I_c has a value of 0.

We validated numerically equation (70) by computing the Laplacian L of the 1000 images from the ECSSD dataset [68], then computing I_c using the Green's function $\check{V}_{mono}^{\mathcal{F}}$. We found the root mean square error (RMSE) to be 0.011 on 256 levels, which is 0.004% of numerical error, which is negligible.

5.2.3.3 A universal convolution reversal?

At first sight, the equations developed in the previous section seems to reverse any convolution kernel K , since equation (66) finds the reverse kernel of any operator. However, the Green's function is only defined for differential operators, meaning that non-differential operators do not necessarily have a reverse.

For example, reversing the popular Sobel gradient operator [23,35] can be done with equation (66), but there will be a significant error on the regions of high gradient. This is because the Sobel operator is a blurred version of the gradient operator, which dissipates high frequencies and cannot be reversed completely. Hence, the resulting image I_c from equation (70) for a Sobel gradient is a blurred version of I .

5.2.3.4 Computation complexity

As shown previously, the Laplacian L is solved using equation (70), where the only operations consist of the Fourier transforms \mathcal{F} and the element-wise product \circ . In this case, the computation complexity will be dominated by the FFT algorithms with a computation complexity of $O(n \log n)$ [23], where n is the total number of pixels.

Although the complexity is not linear, the logarithmic term becomes less important when the number of pixels is near the million, which is typical for images.

5.3 Applications in computer vision

There are many already-proven applications of the Laplacian solvers in computer vision, including seamless cloning, seamless composite, and animated diffusion curves [57,59,61], etc. Those applications are part of a branch called gradient-domain image editing (GDIE) [58].

Since they are already demonstrated, we will only focus mainly on showing the proof-of-concept of the GFC with some comparison to Perez [57], Jeschke [59] and Tanaka [61] methods. Using the development of the previous section, we know that equation (70) is a least-error solver of the Laplacian. We will also demonstrate that the proposed approach is significantly faster than competing methods and that it can be leveraged for machine learning (ML) applications.

5.3.1 Solving the image Laplacian

In this section, we summarize the GDIE process using our proposed GFC method. Then, we benchmark the solver computation time and error against competing methods.

5.3.1.1 GDIE process summary

Figure 5-1 shows a summary of the process used to solve the modified gradient for GDIE applications. All those steps are simple to implement in OpenCV and Matlab since they mostly use already available functions in their respective computer vision toolboxes. Some of the process summary steps, such as the gradient editing and color correction, will be discussed in later sections.

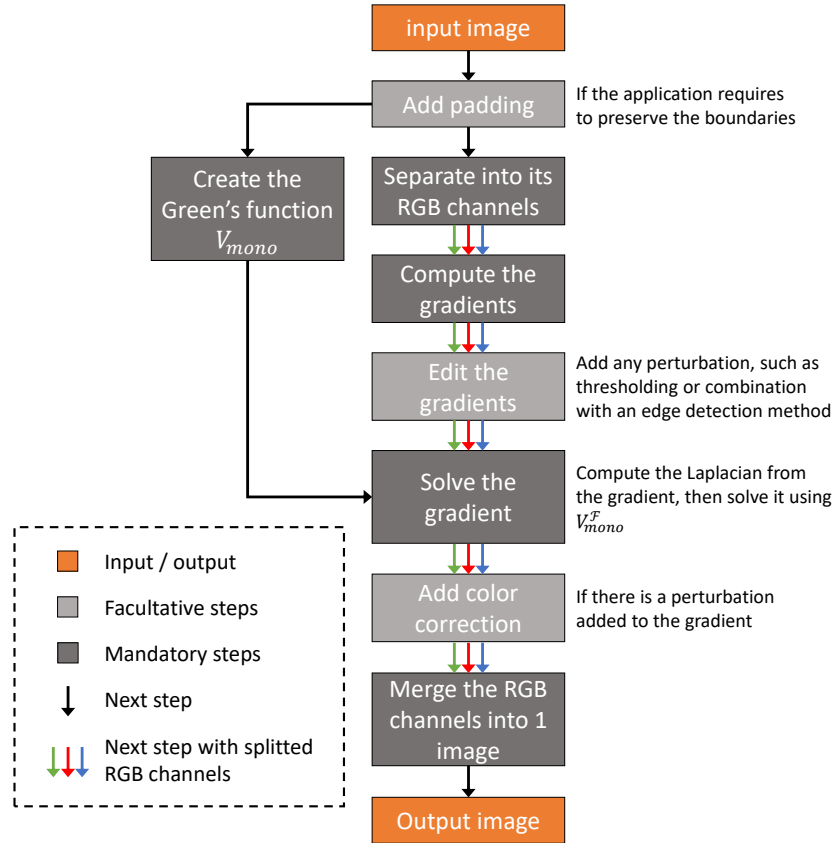


Figure 5-1 : Process summary of the gradient domain image editing.

5.3.1.2 Pseudo-code

In this section, we demonstrate the simplicity of GFC by providing some Python-based pseudo-codes.

First, Algorithm 5-A shows how to compute the Green's function `green_F` for an image of size `image_size` using equation (66). Then, Algorithm 5-B shows how the previous Green's function is used to solve the given Laplacian `padded_L` with equation (70).

For both algorithms, we must keep in mind that the 2D FFT `fft` and inverse FFT `ifft` produce complex outputs, meaning that the products and division must be used accordingly.

The code simplicity allowed us to implement the solver using Matlab, C++ (OpenCV) and Python (Tensorflow and Pytorch).

Algorithm 5-A. Python based pseudo-code for computing Green's function

```

# Inputs:
# image_size: The size of the image
# pad: The padding to add around the image

# Find the size of the desired matrices
pad ← 4
green_function_size ← image_size + 2 * pad

# Create the Dirac and Laplace kernels
dirac ← zeros(green_function_size)
dirac[1, 1] ← 1
laplace ← zeros(green_function_size)
laplace[0:3, 0:3] ← [[0, 1, 0],
                    [1, -4, 1],
                    [0, 1, 0]]

# Compute the Green's function
green_F ← fft(dirac) / fft(laplace)

```

Algorithm 5-B. Python based pseudo-code for solving the padded Laplacian

```

# Inputs:
# padded_L: The Laplacian of the padded image
# green_F: The result of Algorithm 5-A

# Solving the padded Laplacian
I ← ifft(fft(padded_L) * green_F)

# Integration constant and unpadding
I ← I - I[0, 0]
I ← I[pad:-pad, pad:-pad]

```

5.3.1.3 Computation time benchmark

As explained previously, the computation time of our proposed GFC method is low since FFT is highly optimized on CPUs and GPUs [35,36]. For example, the computation time is around 18ms on MATLAB® with an *Intel® i7-6700K* processor for a gray image (single channel) of resolution of 801x1200. Also, using MATLAB's *GpuArray* with the GPU *nvidia® GTX 1080 Ti*, the computation time when the overhead is eliminated is around 0.8ms.

In all the implementations, we used 32-bit floating points, since a double precision is not required.

Our method (GFC). In Figure 5-2, the total time for the GFC is noted 1.3ms, which includes 0.5ms for the preparation such as verifying the parameters and sending the matrices to the GPU. The remaining 0.8ms is used for solving the gradient.

For the GFC method, the computation time in Figure 5-2 does not include the computation of the optimal Green's function $\check{V}_{\text{mono}}^{\mathcal{F}}$ since it can be pre-computed with equation (69). The time to build it is 5ms on the GPU and 36ms on the CPU. Even if $\check{V}_{\text{mono}}^{\mathcal{F}}$ is not pre-computed, the method is still fast enough to out-perform any competing algorithm, since $\check{V}_{\text{mono}}^{\mathcal{F}}$ is computed only once for the 3 channels of an image.

With the logarithmic scale of Figure 5-2, we can observe that the proposed GFC method is orders of magnitude faster than competing algorithms, such as Perez et al. [57] or Jeschke et al. [59].

Our method (GFC) using Pytorch batches. Since one of our objectives is to develop a method compatible with CNN, we decided to implement our method on the Pytorch [117] machine learning library. For the Pytorch implementation, we use batches of 100 different images of size 801x1200, since it is similar to how CNN use batches of features and can be useful for video editing. On a CPU, we found that the batches did not improve the computation time. However, on a GPU, the computation time for a single image (~0.9ms) was almost identical to the batch of 100 images (~1.0ms). Hence, the average time per image is 0.01ms as noted in Figure 5-2. Since 100 images are near the memory limits of our GPU, the 0.01ms per image is the fastest we can achieve in parallel.

Perez method. The Perez [57] algorithm is downloaded from MathWorks [118], and later optimized to use the full capacities of MATLAB, but the matrix inversion alone takes 1770ms with another 1270ms to build the sparse matrix. It has no GPU implementation.

Tanaka method. The Tanaka [61] algorithm is written by the author and is downloaded from MathWorks [119]. The computation time on the CPU to perform the cosine-transforms required to solve the Laplacian is around 292ms with a preparation time of 2ms. Furthermore, an additional time of 85ms is added to compute the cosine-transform solver, but it is not included in Figure 5-2 since it can be pre-computed.

McCann and Pollard multigrid method. Their method could not serve as a benchmark in its current form. Indeed, the provided code is implemented on older hardware and 32-bit libraries, and since their binaries only implement diffusion curves. According to their work [112], their proposed multi-grid solver requires around 10 iterations to converge, so that the process lasts ~110ms for an 801×1200 image on the older GPU *nvidia® GeForce 8600 GTS* with a performance of 93 GFLOPS [120]. On the *nvidia® GTX 1080 Ti* with 11340 GFLOPS [120], the fastest expected time is $0.9\text{ms} \left(\frac{110 \cdot 93}{11340} \right)$. This computation time is only possible if all the CUDA cores are used since the GPU clock is only twice the speed [120]. Hence, the method proposed in this paper, which solves the gradient in 0.8ms (without preparation time), is expected to be equal or faster than McCann for a single image. Furthermore, the proposed method is parallelizable to 100 images in 1 ms, but we do not know if the same is true for McCann.

Jeschke method (diffusion curve only). The Jeschke algorithm is provided with their paper [59], but it is only implemented for diffusion curves. Hence, an ideal comparison with their algorithm is not possible and the time is not included in Figure 5-2. Their algorithm was benchmarked to 6.2 ms on GPU and 476.2 ms on CPU for a single channel computation. Although the comparison is not ideal, this is orders of magnitude slower than our implementations.

Green's function based methods. The methods proposed by Sun et al. [113,114] and Ilbery et al. [115] are both based on Green's function diffusion. However, they are only implemented for diffusion curves and cannot directly work with discrete grids for image editing purposes. This is because they compute the Green's function in a continuous bounded 2D space for application on vector curves. Therefore, a comparison is not directly possible.

Other methods. Other methods such as the one proposed by Bhat et al. do not perform real-time image editing as stated in their paper [58], which means that it is definitely slower than the proposed approach.

In summary, the proposed GFC algorithm runs orders of magnitude faster for discrete images than competing algorithms. Compared to the Tanaka method, the improvement is 16x faster on CPU and 172x faster on GPU. Furthermore, our GPU Pytorch implementation shows that the computation time for batches of 100 images is the same as for a single image. Since the method is fast, we expect that a major part of the running time in a real application will be due to overheads and verifications.

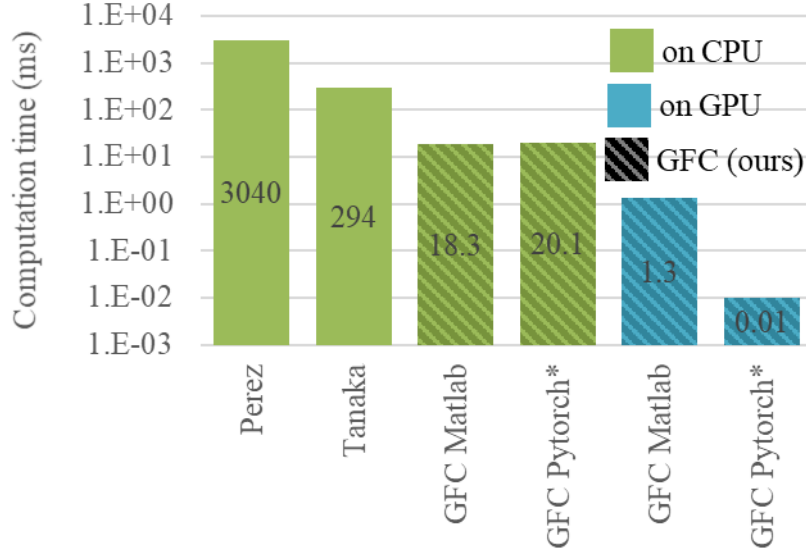


Figure 5-2 : Computation time (ms) in logarithmic scale for a single channel gradient solving of resolution of 801x1200, including the preparation time. The Perez [57] and Tanaka [61] methods have no GPU implementation. The Pytorch* implementation is tested on batches of 100 images, and the total time is divided by 100.

5.3.1.4 Non-conservative solver benchmark

We proved in section 5.2.2 that the Green's function is the least-error solver for any non-conservative fields \mathbf{E}_p . In this section, we demonstrate empirically that our method has less error than the Perez [57] and Tanaka [119] methods.

To demonstrate it, we use the 1000 images from the ECSSD dataset [68] and compute the gradients \mathbf{E} . We modify the gradients by setting any value below a given threshold to 0, with thresholds at 10%, 30% and 50%, resulting in \mathbf{E}_p . Then, we solve \mathbf{E}_p using the different methods and find the new gradient \mathbf{E}_c . Finally, we compute the root mean squared error (RMSE) between the gradients using equation (71).

We observe on Figure 5-3 that the RMSE of our GFC method is consistently lower than competing methods. For the 10% threshold, GFC has an RMSE 16% lower than Tanaka and 76% lower than Perez.

$$\text{RMSE} = \text{mean} \left((E_c - E_p)^2 \right) \quad (71)$$

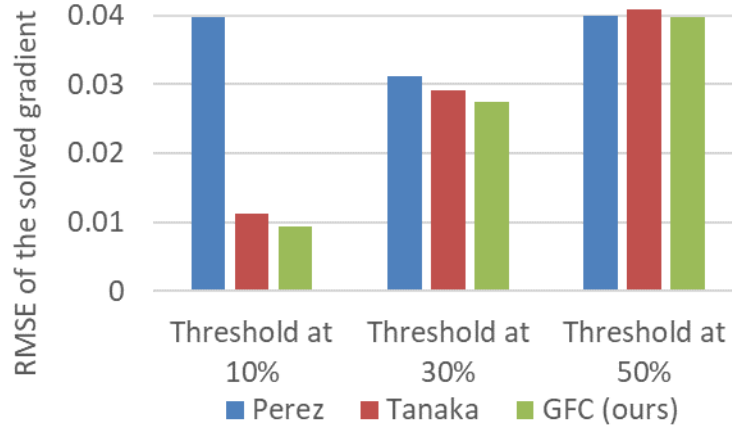


Figure 5-3 : Comparison of the RMSE between E_c and E_p , where the thresholds are the perturbation used to generate E_p . The displayed values are the mean of the RMSE on the 1000 images of the ECSSD dataset [68].

5.3.2 Gradient-domain image editing

From the mathematical proof presented in section 5.2.2, the GFC proved to be the least-error solver for any perturbed gradient. In the case of GDIE, the gradient perturbation is voluntary. It is mainly used for applications such as Poisson blending, diffusion curves [57,59,61] and edge editing [58].

This section will show the performance of the method for Poisson blending, as well as additional possible gradient-domain applications such as the proposed gradient domain merging (GDM) based on the work of Bhat et al. [58]. Those applications can potentially be used in image/video editing software, as well as image pre-processing.

5.3.2.1 Poisson blending

Poisson blending is a type of GDIE that allows merging the gradient of 2 different images, such that the blending is seamless. Since the proposed GFC approach has a low computation time for

large images, as demonstrated in section 5.3.1.3, our implementation of the Poisson blending uses a blend region that is bounded by the total size of the image. This means that if the cropping region passes through a high gradient region, our method is better at compensating the error. This is shown inside the blue circle of Figure 5-4 where the GFC approach solves smoothly the cropped edge. Also, the GFC blending appears more natural since the left side of the stamp is more transparent.

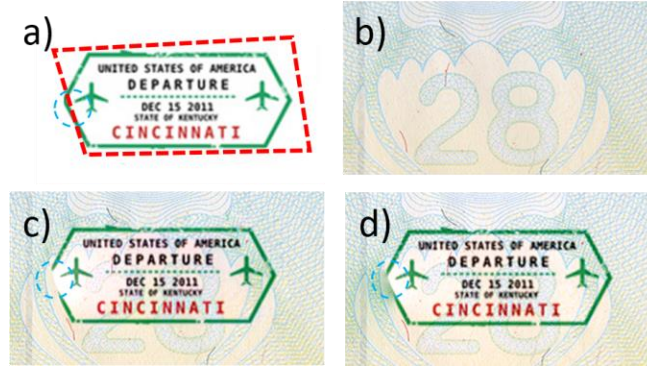


Figure 5-4 : Example of Poisson Blending application; (a) Stamp to copy, with the red-dotted lines being the cropping region and the blue dotted circle being a region of the stamp that is accidentally cropped; (b) destination image; (c) Poisson blending from Perez algorithm [57,118]; (d) Proposed GFC blending.

In other cases where the cropping region does not pass through a high gradient, the results of the proposed GFC method is identical to the Perez method.

5.3.2.2 Preserving the coloration

The equation (70) presented an optimal Laplacian solver in the Fourier domain. In comparison, the literature proposes mostly iterative methods on the Laplacian [57,59,61], which gives an advantage for our method by making it faster and easier to implement. However, computing the Laplacian from the perturbed gradient requires an additional computing step to preserve the brightness and contrast.

The problem when editing the gradient in an image is that the desired potential is not necessarily the result given by V_E since we want to preserve the color information. Hence, we define a new corrected potential $I_{c,corr}$ in equation (72), where σ indicates the standard deviation and the top bar “ $\bar{}$ ” indicates the average. As stated in section 5.2.3.2, it is possible to add any constant to I_c without changing the validity of the equation, which means that the addition and subtraction of

equation (72) do not affect the potential. For the σ ratio, it is meant to preserve the initial contrast of the image, and it simply changes the norm of the gradient by a constant factor.

$$I_{c,corr} = (I_c - \bar{I}_c) \frac{\sigma(I)}{\sigma(I_c)} + \bar{I} \quad (72)$$

In case no perturbation is added to the gradient, we have $I_c \approx I$. This means that almost no correction will be added to I_c , resulting in $I_{c,corr} \approx I_c$.

5.3.3 Gradient thresholding

Using the color preservation of equation (72), Figure 5-5 shows an of thresholding the gradient at 10% of the highest possible gradient and computing the solving the new image with equations (70) and (72). We can see that most features of the initial image are preserved, but that there are less texture and fine elements.

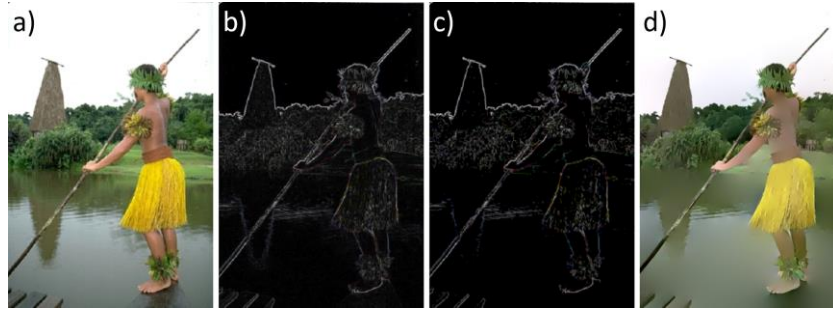


Figure 5-5 : Example of gradient thresholding and solver steps. (a) Original image; (b) Gradient $|E|$; (c) Thresholded gradient at 10%; (d) Solved image $I_{c,corr}$.

Figure 5-6 shows 2 more examples of GDIE with a 10% gradient thresholding. In those images, the castle reflection is completely erased, along with the clouds. For the leopard picture, almost all the background information is erased except for the leopard.

The differences between our proposed Laplacian solver GFC and the one proposed by Perez [57] are negligible in the case of gradient removal. Hence, we do not present comparison images since the differences are imperceptible to the human eye.

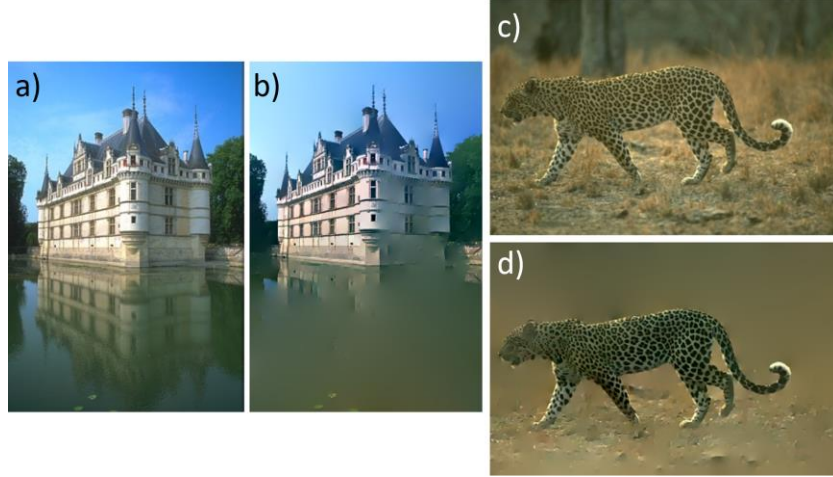


Figure 5-6 : Examples of solved images $I_{c,corr}$ after the gradient threshold at 10%. (a) Image of a castle; (b) Solved castle image after 10% gradient threshold; (c) Image of a leopard; (d) Solved leopard image after 10% gradient threshold.

5.3.4 Gradient domain merging (GDM)

In this section, we present a method of editing an image by merging edges information with gradient information.

A similar approach was used by Bhat et al. [58], which computed the salient gradient map to enhance the original gradient via cosine similarity. What we propose instead is to use the edges produced by ML algorithms and merge them to the gradient via a geometric average.

The motivation of using machine learning edges prediction is that we believe future work could benefit from implementing the Green's function inside ML algorithms. A simple example would be to enhance the contrast of the important objects via GDM, thus making it easier for the ML method to detect the object.

The proposed GDM approach consists of combining the gradient with the edge information using a weighted geometric average defined in (73). The product enhances the gradient where edges are present but reduces them where edges are not present. In the equation, \mathbf{E}_p is the perturbed gradient, \mathbf{E} is the original gradient, C is the intensity of the edge detection, \circ is the element-wise product, and $\alpha = [0,1]$ is the weight associated to the geometric mean. Also, the orientation of the perturbed gradient θ_{E_p} is equal to the orientation of the original gradient θ_E . A higher α attributes more weight to the edges, while a lower α attributes more weight to the gradient.

$$|E_p| = |E|^{1-\alpha} \circ C^\alpha$$

$$\theta_{E_p} = \theta_E$$
(73)

5.3.4.1 Contrast enhancement between objects

Figure 5-7 shows the different steps involved in the GDM method. The perturbed field E_p is produced by equation (73) and solved by equation (72). We observe that the GDM produces a loss of texture, but that the contrast between the objects are enhanced.

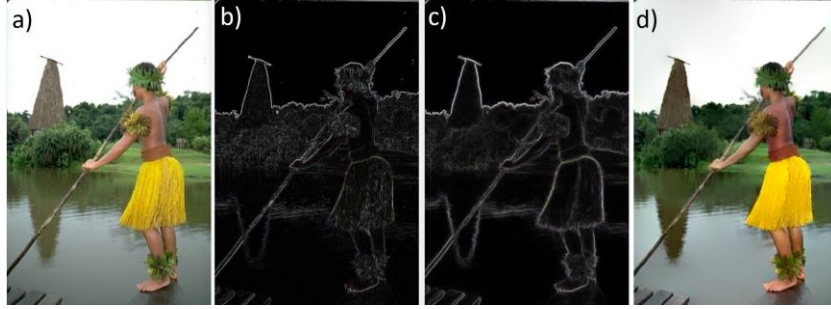


Figure 5-7 : Examples of the steps involving the GDM equation (73). (a) Original image; (b) $|E|$: Gradient; (c) $|E_p|$: Gradient merged with the SE method [64] and $\alpha = 0.5$; (d) Solved image.

In Figure 5-8, we can see the effect of using different α parameters. The higher the parameter α is chosen, the stronger is the contrast between objects. However, a higher α creates discoloration in the image. This is because a higher α produces a field that is too different from the original field, which yields in undesired coloring and brightness artefacts.

In Figure 5-9, we can observe more examples of GDM using $\alpha = 0.5$. We see that the deep learning edges RCF [25] produces higher contrast than the between objects than the random forest SE edges [64].

In Figure 5-10, we can observe that Bhat [58] method of saliency sharpening enhances the folds of the clothing and the lines in the background. This is opposite to our method which reduces the folds and the background texture but enhances the colors of the foreground objects. This demonstrates that our method is fundamentally different than previously proposed edge enhancement methods and should not be used for the same purposes.

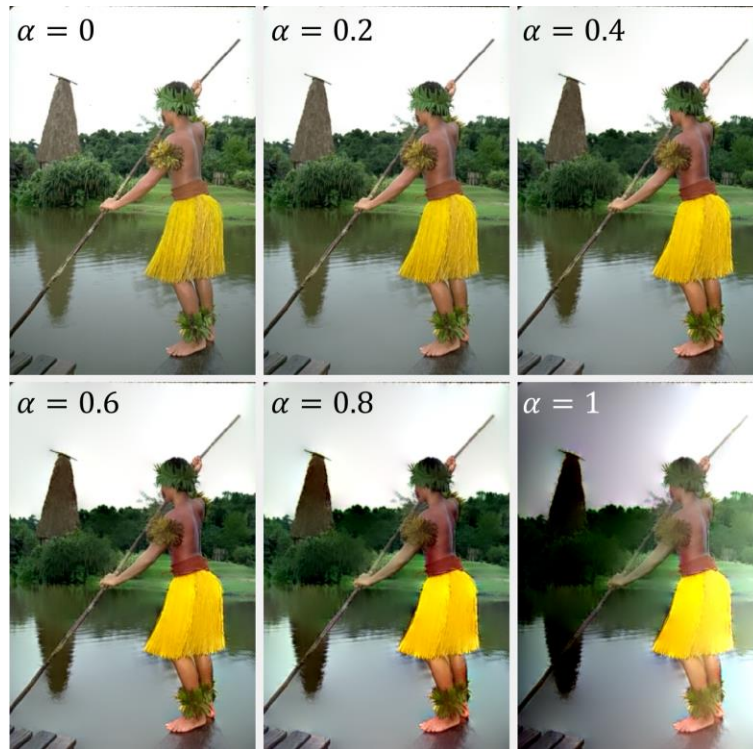


Figure 5-8 : Example of GDM using equation (73) with a random forest edge detector [64] and varying parameter α .

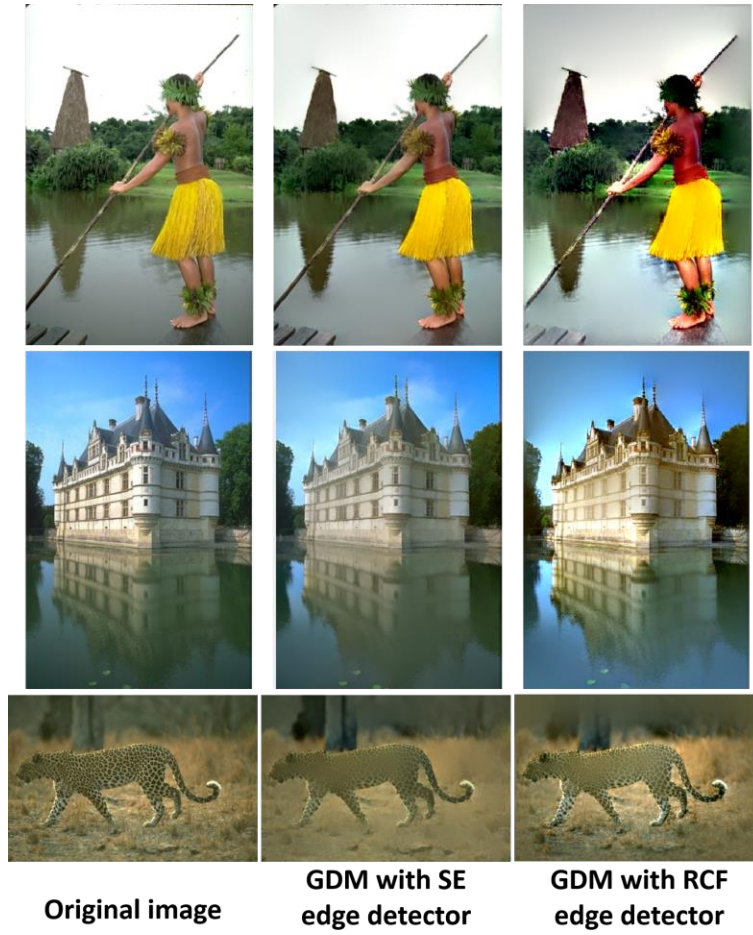


Figure 5-9 : Examples of solved images $I_{c,corr}$ with a perturbed gradient from equation (73) with edges information from SE method [64] and RCF method [25] and $\alpha = 0.5$.

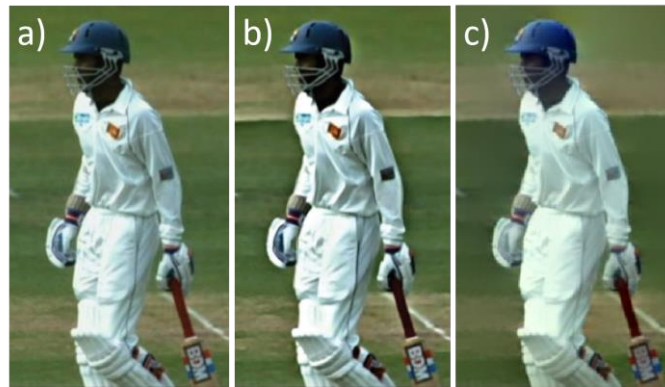


Figure 5-10 : Comparison of edge merging methods. (a) Original image; (b) Saliency sharpening from Bhat et al. [58]; (c) Enhancement using our GDM method with $\alpha = 0.5$ and SE edge detector.

5.3.4.2 Painting effect using thin edges

For the GDM method, it is also possible to thin the edges before merging them with the gradient. This thinning is often called non-maximal suppression (NMS) and is natively implemented in some edge detection such as SE [64]. Applying NMS to the edges removes almost completely the texture information, meaning that the solved image resembles a painting, as observed in Figure 5-11.

Since the thin edges \mathcal{C} do not necessarily intersect the gradient, we thicken the gradient \mathbf{E} by using a Gaussian filter with a standard deviation $\sigma = 1$.

In Figure 5-11, we compare our GFC method and the one proposed by Perez [57,118]. First, we notice that the GFC approach has better color preservation than the Perez method. For example, we observe on the person image that the sky has a gradient of different colors. We also observe that the castle image has many small coloration artifacts inside the castle and at the top of the sky. For the leopard image, both methods yield similar results. Thus, the proposed GFC method produces a more natural painting effect since it is more accurate on fine details and color restoration than the competing Perez algorithm [57].

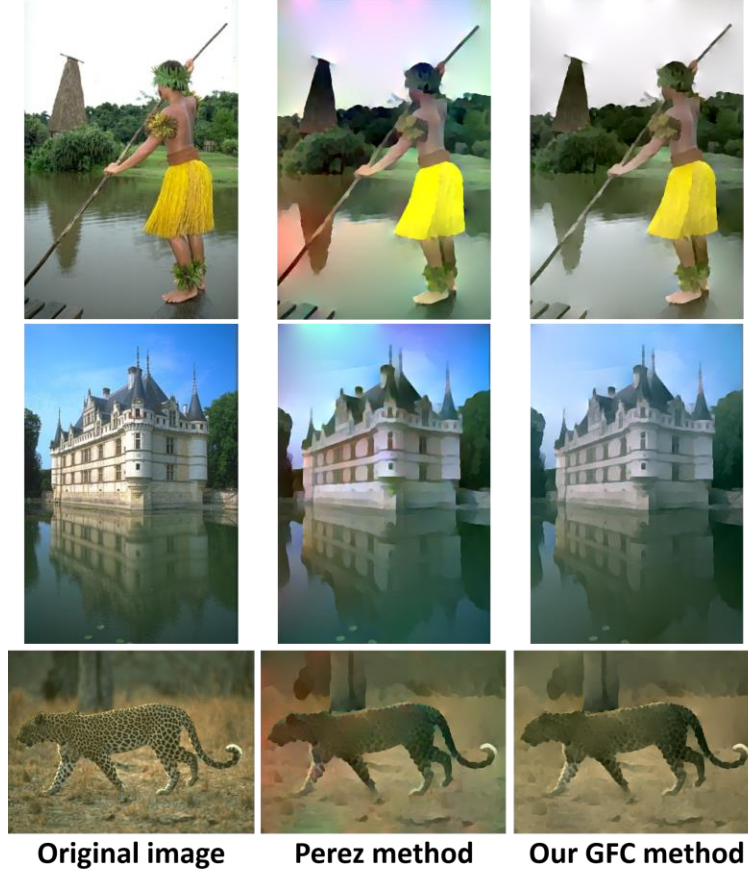


Figure 5-11 : Examples of solved images $I_{c,corr}$ with a perturbed gradient from equation (73) with edges information from SE method [64] with NMS and $\alpha = 0.5$.

5.4 Future work

In this section, we briefly discuss possible future work of our research concerning the convolutional neural network (CNN) applications and tensor processing unit (TPU) implementation.

Machine learning applications. As shown in Figure 5-2, one of the advantages of our method is that the computation time is orders of magnitude faster than competing methods, and 100 folds parallelizable. Furthermore, Algorithm 5-B shows that the code is very simple to implement if a 2D FFT is available. These advantages will allow future work to use the GFC inside CNN, thus allowing the networks to natively learn gradient-domain image editing. In fact, our Pytorch and Tensorflow implementations of GFC can be easily integrated inside a network since the backward propagation of the FFT is natively available.

TPU implementation. To further improve the computation speed and the parallel processing of our method, it will be interesting to implement it on TPU. These new processors allow parallelizing more operations by using 16 bits floating points, which can heavily benefit the computation of FFT [121].

5.5 Conclusion

This study detailed the development of the GFC method, which allows solving any field or Laplacian for gradient domain image editing purposes. First, we explained the theory behind the Green function convolution (GFC), and we mathematically proved in section 5.2.2 that it is the least error solver. Then, we demonstrated empirically on 1000 images that the RMSE error is negligible with a value of 0.004%. Moreover, Figure 5-3 also showed that the solver error on non-conservative fields is consistently lower than competing methods. Figure 5-2 also showed that the method is almost instantaneous with a computation time of 1ms for the parallel processing of 100 images with resolution 1200x801. Finally, we demonstrated different use-cases of gradient domain image editing and introduced GDM, the first method of merging learned edges with gradients for texture removal and contrast enhancement.

In summary, this study allowed to build a robust and fast way to edit an image from its gradient which can be used in many applications. The code is fast enough to have a negligible impact on the computation time and simple enough to be implemented in any language. Future works could focus on more concrete applications, such as supervised image/video editing and machine learning applications.

Acknowledgment

The authors are grateful to NSERC, through the discovery grant program RGPIN-2014-06289, and FRQNT/INTER for their financial support.

CHAPTER 6 ARTICLE 4: SALIENCY ENHANCEMENT USING GRADIENT DOMAIN EDGES MERGING

Dominique Beaini, Sofiane Achiche, Alexandre Duperré, Maxime Raison

Polytechnique Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

Submitted to Image and Vision Computing, Elsevier, December 2018

Abstract

In recent years, there has been rapid progress in solving the binary problems in computer vision, such as edge detection which finds the boundaries of an image and salient object detection which finds the important object in an image. This progress happened thanks to the rise of deep-learning and convolutional neural networks (CNN) which allow extracting complex and abstract features. However, edge detection and saliency are still two different fields and do not interact together, although it is intuitive for a human to detect salient objects based on its boundaries. Those features are not well merged in a CNN because edges and surfaces do not intersect since one feature represents a region while the other represents boundaries between different regions. In the current work, the main objective is to develop a method to merge the edges with the saliency maps to improve the performance of the saliency. Hence, we developed the gradient-domain merging (GDM) which can be used to quickly combine the image-domain information of salient object detection with the gradient-domain information of the edge detection. This leads to our proposed saliency enhancement using edges (SEE) with an average improvement of the F-measure of at least 3.4 times higher on the DUT-OMRON dataset and 6.6 times higher on the ECSSD dataset when compared to competing algorithms such as denseCRF and BGOF. The SEE algorithm is split into 2 parts, SEE-Pre for preprocessing and SEE-Post pour postprocessing.

Keywords: Computer vision, Salient object detection, Saliency map, Gradient-domain editing, Edge detection.

6.1 Introduction

Recent years have seen great progress in solving binary problems in computer vision, such as edge detection [11,25,66] and saliency [11,26,76]. Saliency methods used different approaches based on multiple features, such as clustering and density [69–71], concavity [72], contrast filtering [73],

and background detection [74], but did not use edges since there exist no methods of merging edges with saliency. Furthermore, since the arrival of convolutional neural networks for saliency detection around the year 2015 [87], many highly effective algorithms were proposed such as MDF [87], DCL [76] and DSS [11,26]. The CNN was also used for edge detection such as RCF [25], with a work by Hou et al. in which they propose a unified framework to compute both saliency and edge detection [11], but without combining both results together.

Although there have been much research works on both saliency and edge detection, only a few propose to improve the saliency using edges [77–79]. Other works propose to use background detection [74], contrast enhancement and texture smoothing [80] to improve the results, but as benchmarked by Patel et Raman [77], those algorithms do not work well on the recent CNN models, since CNN models are more performant at detecting those features. In fact, only the denseCRF [79] and the boundary-guided BGOF [78] method are proven to improve the performance of CNN-based saliency detection [78]. This is supported by the simple fact that the saliency should be low outside the boundary and high inside it, which is exactly what denseCRF and BGOF work try to optimize. However, they rely on an energy minimization of region segmentation instead of edge-based boundary conditions.

Hence, our objective is to develop a method that merges the results of edge detection algorithms with the results of saliency detection algorithms to improve the performance of the saliency maps. For this objective, we propose saliency enhancement using edges (SEE) which allows one to merge the results from top saliency methods with top edge detection methods.

Merging the region saliency maps with the edge maps is complex since they typically do not intersect, meaning that they cannot be combined by standard operations such as additions and multiplications. To solve this problem, our previous work showed that we can use the numerical 2D Green’s function as convolutional kernels to extrapolate edges into surface information [15–17,21]. These convolutions allowed to quickly evaluate the probability of being inside a given contour [16] and to smooth an image according to the edges given by an edge detection technique [17]. The SEE approach proposed in the current paper assumes that the gradient of the saliency maps is similar in nature to the edges map. Hence, it uses the gradient-domain to merge those features, then it solves the gradient using a numerical Green’s function convolution [17]. The SEE method is split into 2 parts: SEE-Pre to preprocess the image based on the edges and the SEE-Post

to post-process the saliency map based on its edges. The optimal results occur when those 2 parts are used together.

6.2 Saliency enhancement using edges

For the saliency enhancement using edges (SEE) method, we propose to merge edges with the saliency and the image using gradient-domain merging (GDM). This SEE method is separated into 3 steps, the SEE-Pre which preprocesses the image, the SEE-Post which post-processes the image and the contrast enhancement.

The SEE-Pre works by blurring the non-salient regions and enhancing the contrast of the salient regions. Competitive approaches with the same goal were proven to work on standard saliency models [77–79], but not on the newest deep-learning algorithms [78]. The SEE-Post works by keeping only the salient regions that are bounded by salient edges, similar to the most efficient available method BGOF [78].

The following sections will first explain the SEE method and will follow by detailing each step: the salient edge detection, the GDM, the SEE-Pre, the SEE-Post, and the contrast enhancement.

6.2.1 The complete SEE method

An overview of the SEE method can be seen in Figure 6-1 where the edges and the salient edges C_0 and the image I_0 serve as the 2 inputs of the method. With the regular saliency being S_I , we observe that the SEE-Pre is first applied to get the enhanced saliency $S_{I_{0R}}$, followed by SEE-Post to obtain $S_{I_{0RR}}$. Finally, the saliency map is normalized in the range $[0, 1]$ and the contrast is enhanced to get the saliency $S_{I_{0RC}}$.

We can observe in Figure 6-1 that the precision/recall curve is almost perfect for the given example, with a precision P near 100% at every point where the recall is R is lower than 94%. This shows in the current example that the SEE method can significantly improve the resulting saliency map when both the preprocessing SEE-Pre and postprocessing SEE-Post methods are used together. More details on the precision/recall curve is given in section “6.2.5 Evaluation datasets and metrics”.

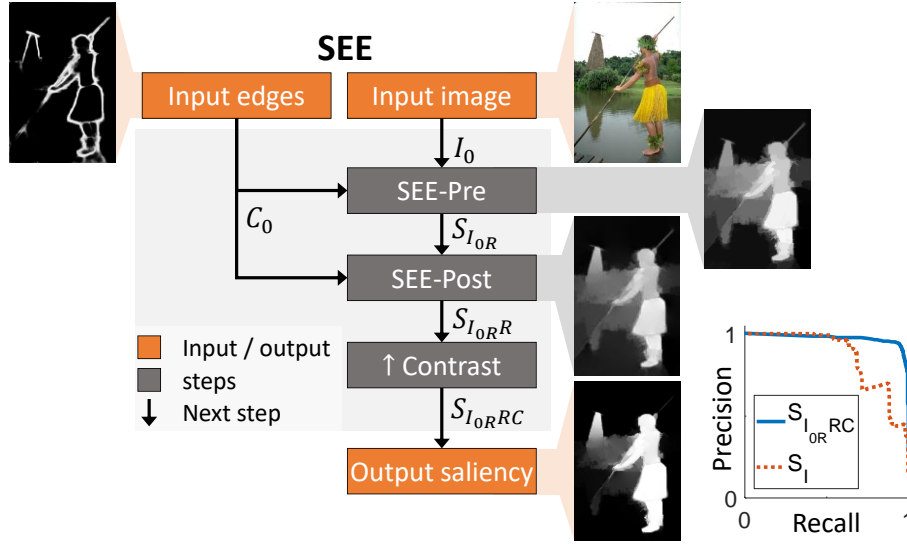


Figure 6-1 : Diagram of the complete SEE method, which allows improving the saliency via a combined image preprocessing and saliency map postprocessing. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example where S_I is the original saliency and $S_{I_{0R^RC}}$ is the same saliency improved using our SEE method.

6.2.2 Salient edge detection

Before merging the edges in the gradient domain for our SEE approach, we must first understand which edges should be extracted from the image. Most edge detection methods focus on extracting all the edges from an image and its background [11,25,64] by basing their training on the BSDS500 dataset [49]. In the case of saliency improvement, this yields to undesired edges that are not useful for the saliency improvement.

For the task of salient object detection, the most used dataset for the training is MSRA10K with 10,000 images [122] or its earlier version MSRA-B with 5000 images [67]. Those saliency-based datasets include a ground-truth G image with value 1 at every pixel inside the salient object and 0 at every pixel outside the salient object. By taking the boundaries of those G images for the salient objects, we get a new ground-truth G_{sal} for the salient edges. Using the new G_{sal} , we can retrain the edge detection models to detect only the salient edges.

We tried to use 2 different edge detection models to detect the salient edges by retraining or fine-tuning them with the G_{sal} . The first method, SE, uses random structured forest for learning [64,65]

and it was one of the best method available before the introduction of the CNN [25,65]. The second method, RCF [25], uses deep CNN for learning [25], and it is one of the best edge detection method available [11,25]. Examples of results are shown in Figure 6-2. The training models on the BSDS500 were already available for both methods. For the MSRA10K, we retrained the SE method with 10k iterations and we fine-tuned the RCF method with 1k and 5k iterations. The SE method had only moderate success at eliminating non-salient edges, and with many missing salient edges. The finetuned RCF method with 1k iterations and 256 images per iteration performed a lot better by eliminating the non-salient edges and enhancing the salient edges. The same method with 5k iterations appears a lot cleaner, but also eliminates the salient edges which is undesirable.

A full test of the SEE method with a different number of iterations of the RCF method suggest that 1k iterations of finetuning with 256 images per iterations are near optimal results for saliency improvement. Hence, we used this finetuning of RCF for the rest of the paper, which we will denote as RCF-MSRA10k-1k. Since the dataset has 10,000 images and that we used 8000 for training, each image was trained an average of 32 times, which is low enough to avoid an overfit.

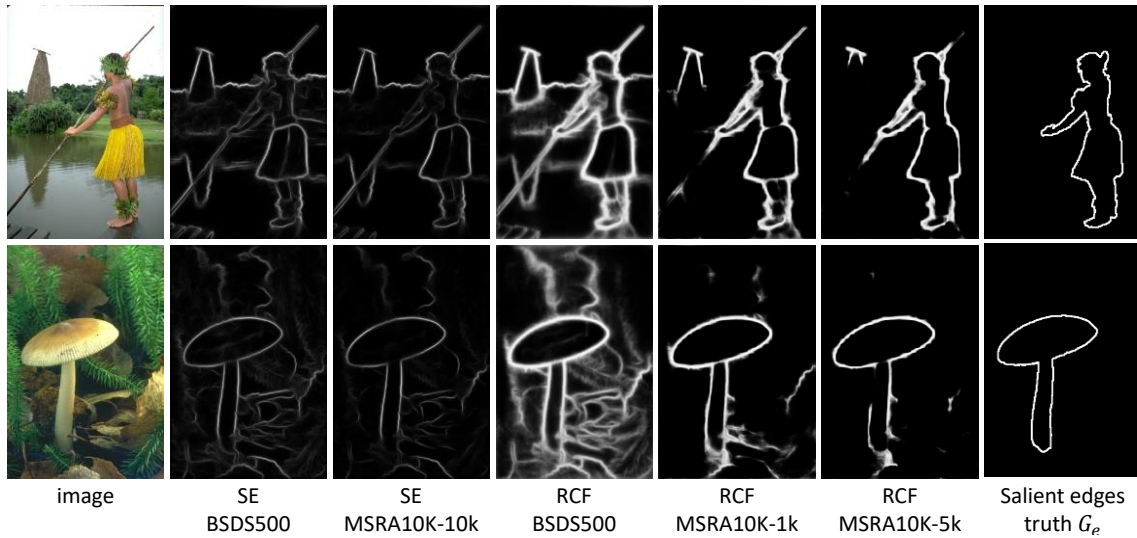


Figure 6-2 : Comparison of SE and RCF edge detection methods trained or finetuned on the BSDS500 or MSRA10K datasets, with the number of iterations of retraining, using 2 example images from the ECSSD dataset. The salient edges will be used as inputs for the proposed SEE method.

6.2.3 Gradient-domain merging

Our previous work demonstrated that an image can be reconstructed from its gradient using a simple Green's function convolution (GFC) [16,17]. Since the convolution converges to the optimal solution even when a perturbation is added to the gradient, it showed that it can be used for robust edge-preserving blurring with any edge-detection method [17]. The blurring was done by gradient-domain merging (GDM) with the detected edges, followed by a Green's function convolution that computes the image associated to the modified gradient.

The current section presents the main equations that are used for gradient domain merging (GDM), which allows merging the results of edge detection with an image or saliency map. All the details for the equations can be found in previous work [15–17,21], with the most similarities to our previously proposed Gradient and Laplacian solver [17]. We make use of complex numbers to avoid redundant equations and convolution kernel by representing the x and y axes in the same equation. However, in practice, those equations can be separated into 2 different equations representing the real and imaginary parts.

Since the GDM requires its inputs to be in the gradient domain, the first step is to compute the gradient \mathbf{E} of the image I_0 and its orientation θ using the complex right-derivative kernel $K_{I \rightarrow E}$ in equation (74), where i is the imaginary number. It is possible to use 2 real kernels instead of $K_{I \rightarrow E}$, which would require 2 convolutions with the real/imaginary part of the kernel [16,17]. It is important to note that a zero-valued padding of 3 pixels must be added to I_0 to ensure continuity of the gradient and Laplacian at the border of the image [17].

$$\begin{aligned}
 K_{I \rightarrow E} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & i - 1 & 1 \\ 0 & -i & 0 \end{bmatrix} \\
 \mathbf{E} &= I_0 * K_{I \rightarrow E} \\
 E_x &= \Re(\mathbf{E}) , \quad E_y = \Im(\mathbf{E}) \\
 |\mathbf{E}| &= \sqrt{E_x^2 + E_y^2} , \quad \theta_E = \text{atan}\left(\frac{E_y}{E_x}\right)
 \end{aligned} \tag{74}$$

Using a merger function for the norm M_E and another merger function for the orientation M_θ , it is possible to merge the gradient \mathbf{E} with the modified contour C_1 , as seen in equation (73). This allows to obtain the modified gradient \mathbf{E}_p with its orientation θ_p in the complex domain.

$$\begin{aligned} |\mathbf{E}_p| &= M_E(C_0, \mathbf{E}) \\ \theta_p &= M_\theta(C_0, \mathbf{E}) \\ \mathbf{E}_p &= |\mathbf{E}_p| e^{i\theta_p} \end{aligned} \tag{75}$$

Once the modified gradient \mathbf{E}_p is computed, it is mandatory to go back from the gradient domain to the potential (or image) domain. Hence, we need to solve the gradient using the GFC method that we developed in previous work [17]. The GFC method is used since it is fast with around 2 ms of computation time for an image of size 800×1200 and it is optimally robust against perturbations or modifications [17].

Using the newly modified gradient \mathbf{E}_p , we can compute the modified Laplacian L_p . The Laplacian is used since it is more straightforward to solve the Laplacian than the gradient using the GFC method [17]. The complex kernel $K_{E \rightarrow L}$ from equation (76) is used to compute the Laplacian L_p in equation (77), where \mathcal{R} is the real part of the complex value.

$$K_{E \rightarrow L} = \begin{bmatrix} 0 & -i & 0 \\ -1 & i+1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{76}$$

$$L_p = \mathcal{R}(\mathbf{E}_p * K_{E \rightarrow L}) \tag{77}$$

The GFC method states that the Laplacian can be easily solved using the 2D Green's function. We first need to implement the matrices \check{K}_{∇^2} and $\check{\delta}$ using equation (78), which are the zero-padded Laplacian and Dirac's kernels [17]. Then, we compute the optimal Green's function in the Fourier domain $\check{V}_{\text{mono}}^{\mathcal{F}}$ using equation (79), with \mathcal{F} being the Fourier transform [17]. The value of $\check{V}_{\text{mono}}^{\mathcal{F}}$ can be pre-computed to reduce the computation time of the Laplacian solver.

$$\check{K}_{\nabla^2} \equiv \begin{bmatrix} 0 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & & \\ 0 & -1 & 0 & & \\ \vdots & & & \ddots & \\ 0 & \underbrace{\hspace{10em}}_{size(I)} & 0 \end{bmatrix} \quad \check{\delta} \equiv \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & \\ 0 & 0 & 0 & & \\ \vdots & & & \ddots & \\ 0 & \underbrace{\hspace{10em}}_{size(I)} & 0 \end{bmatrix} \quad (78)$$

$$\check{V}_{\text{mono}}^{\mathcal{F}} = \frac{\mathcal{F}(\check{\delta})}{\mathcal{F}(\check{K}_{\nabla^2})} \quad (79)$$

Finally, the Laplacian can be solved with a convolution in the Fourier domain [17,33] using equation (80), where \mathcal{F} and \mathcal{F}^{-1} are the Fourier transform and its inverse, \mathcal{R} is the real part of a complex number and \circ is the element-wise product. The resulting I_R has a constant value of $-c$ on all of its borders, which will be set to zero by defining adding the integration constant c to I_R [17]. Also, the resulting I_R is cropped to match the initial size of I_0 before the padding.

$$I_R = \mathcal{R} \left(\mathcal{F}^{-1} \left(\mathcal{F}(L_p) \circ \check{V}_{\text{mono}}^{\mathcal{F}} \right) \right) + c \quad (80)$$

A diagram representing the previous steps is presented in Figure 6-3, where C_0 is the salient edges and I_0 is either the saliency map or the RGB image. In the current work, C_0 is computed via the method RCF-MSRA10k-1k. The gradient step is computed with (74), the Laplacian with (77) and the GFC with (80). The preparation steps are flexible and the merging functions M_E and M_θ (73) vary according to the desired application and they will be explained in later sections.

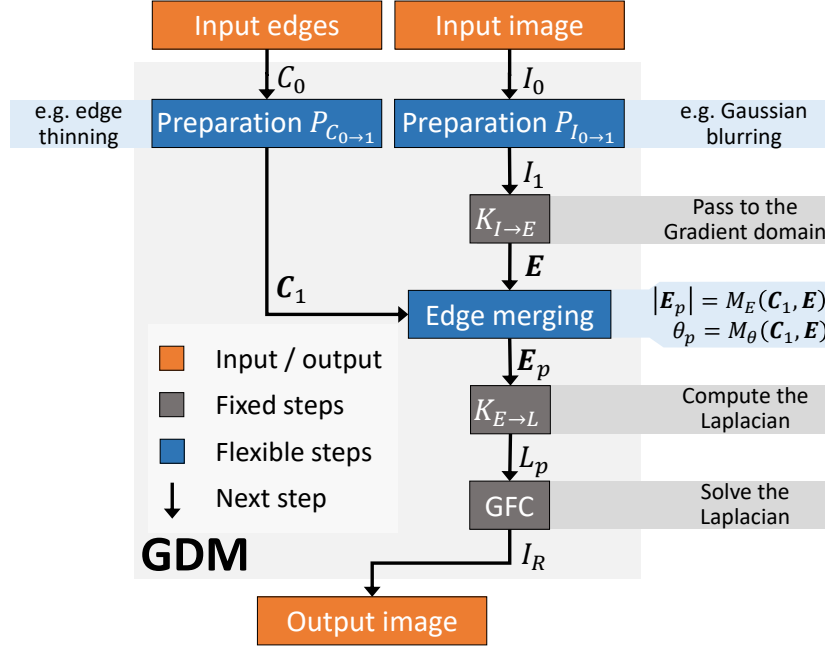


Figure 6-3 : Diagram of the core GDM function, allowing to merge image/saliency information with edges information by passing to the gradient domain and solving the Laplacian equation to go back to the image domain.

In summary, this section explained how to compute the GDM by solving the modified gradient E_p using the Laplacian L_p (77), the pre-computed Fourier-domain Green's function (79), and the Fourier-domain convolution (80). This allows to merge the image or saliency information with the salient edges by using the gradient domain, as shown in Figure 6-3. As stated in our previous work, this computation is fast and easy to implement [16,17], since most computer vision libraries such as MATLAB [36] and OpenCV [35] implement fast Fourier transform (FFT) algorithms. Also, the FFT algorithms are also available on the graphical processing unit (GPU) which allows for faster computation. The estimated computation time for a single channel 400x400 image is 1.5ms when eliminating overhead on the GPU @Nvidia GTX 1080-Ti.

6.2.4 The SEE method

The GDM method explained in the previous sub-section can be used at different stages for the improvement of this saliency, with some variations outlined by the “flexible steps” in Figure 6-3. This section explains how GDM can be used for saliency enhancement using edges (SEE). First, we show how GDM is used for postprocessing of the saliency map (SEE-Post) and preprocessing of the image (SEE-Pre). Then, we show how to combine them into the general SEE method.

6.2.4.1 The SEE-Post method

The first part of the SEE method consists of the postprocessing of the saliency map SEE-Post, which tends to enhance its intensity inside the salient edges and to reduce it outside of them.

Since the SEE-Post acts as postprocessing of the saliency result, then the input I_0 of the GDM is replaced by the saliency S_I computed on the image I . The input C_0 of the GDM is the salient edge computed using RCF-MSRA10k-1k. The output of GDM is the reconstructed saliency S_R . Finally, the SEE-Post method averages S_I and S_R , resulting in the final output S_{IR} . The output S_{IR} combines the higher precision of S_R with the higher recall of S_I for a better overall result. The entire procedure is explained in the Figure 6-4, where is the example saliency S_I is given by the DRFI method [67]. The final results are given later in the section 6.3 and the precision-recall curves are in the section 6.4.

Although our SEE-Post uses different saliency methods, DRFI is chosen for the examples since it allows to better visualize the improvement of the saliency map. In Figure 6-4, we observe how the intensity of the background is significantly reduced while the intensity of the salient person is enhanced. The improvement is shown in the precision/recall curve of Figure 6-4 where the S_{IR} curve has better or equal precision than S_I at every point and S_{IR} is shown have an almost steady precision for any recall value.

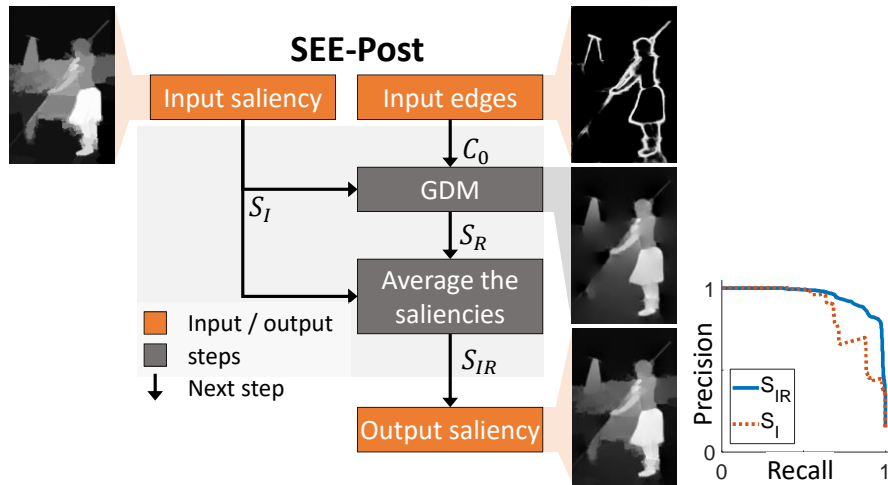


Figure 6-4 : Diagram of the SEE-Post method, which allows to post-process a saliency map based on the salient edge detection. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example.

Now we need to understand the preparation steps for the edges $P_{C_0 \rightarrow 1}$ and for the saliency $P_{S_0 \rightarrow 1}$ that are used in order to obtain those results. For optimal results, we want the edges to be as thin as possible to avoid blurring the edges of the saliency map and to have a more accurate tracing. Hence, the $P_{C_0 \rightarrow 1}$ applies the non-maximal suppression (NMS) algorithm provided with the SE edge detection [64,65] on their @GitHub [123]. However, edges too thin will not necessarily intersect with the gradient, which means that $P_{C_0 \rightarrow 1}$ adds a dilation after the thinning using the *imdilate* function in MATLAB® with a disk of 3-pixels diameter.

Furthermore, the $P_{S_0 \rightarrow 1}$ adds a blurring to the saliency map to ensure the smoothness of the reconstruction, as proposed by our previous work for the painting effect [17]. We use a Gaussian blur with standard deviation $\sigma = 3$ pixels to compute S_1 , which helps ensure a smooth intersection between the gradient of S_1 and the thinned edges C_1 .

The next step is to define the functions M_E^{Post} and M_θ^{Post} from equation (73) to determine how the gradient merging is done. First, we define the merger function for the norm M_E^{Post} according to equation (81), which is inspired by the edge contrast enhancement and painting effect from our previous work [17]. The element-wise product “ \circ ” allows to eliminate every gradient that is not nearby a thin edge, while the square-root allows to preserve $|\mathbf{E}|$ in case $|\mathbf{E}| = C_1$.

$$|\mathbf{E}_p^{\text{Post}}| = M_E^{\text{Post}}(C_1, \mathbf{E}) = \sqrt{C_1 \circ |\mathbf{E}|} \quad (81)$$

Then we define the merger function for the orientation M_θ^{Post} according to equation (82), which always returns the value of θ_C , but shifts it by π if the projection of \mathbf{E} on C_1 returns a negative value. θ_C is defined as the orientation perpendicular to the thin edges. This is based on the fact that our previous work [16] proved that the orientation of dipoles must be perpendicular to the contour C_1 to give the probability that any pixel is inside the given C_1 , which is closely related to the saliency. Furthermore, the shift by a value of π is due to the 2 different possible orientation of dipoles, which must ideally be optimized [16]. Since the optimization is computationally heavy, choosing the orientation in the same direction as \mathbf{E} yields to satisfactory results.

$$\theta_p^{\text{Post}} = M_\theta^{\text{Post}}(C_1, \mathbf{E}) = \begin{cases} \theta_C & \cos(\theta_C - \theta_E) \geq 0 \\ \theta_C + \pi & \cos(\theta_C - \theta_E) < 0 \end{cases} \quad (82)$$

In summary, the general steps for SEE-Post method are presented in Figure 6-4, with the specific flexible GDM steps presented in the list below.

- Contour preparation step $P_{C_0 \rightarrow 1}$
 - Control thinning using NMS, followed by dilation with a disk of 3-pixel diameter.
- Saliency preparation step $P_{S_0 \rightarrow 1}$
 - Smoothing using a normalized kernel Gaussian kernel with $\sigma = 3$
- Norm merging function M_E^{Post} defined in equation (81).
- Orientation merging function M_θ^{Post} defined in equation (82).

6.2.4.2 The SEE-Pre method

In addition to the postprocessing method, the proposed SEE approach also offers a preprocessing method SEE-Pre for the improvement of the computed saliency map. It works by using salient edges to generate a new image where most of the background is eliminated, which helps the saliency method generate more accurate saliency maps.

Since the SEE-Pre acts as a preprocessing of the input image, then the input I_0 of the GDM is the original image I_0 and the input C_0 of the GDM is the salient edge computed using RCF-MSRA10k-1k. The output of GDM is the reconstructed image I_R . Then the SEE-Post method runs twice the saliency algorithm to obtain S_{I_0} and S_{I_R} and averages both outputs, resulting in the final saliency $S_{I_{0R}}$. The averaging allows to merge the better object recall of S_{I_R} with the better boundary precision of S_{I_0} . The entire procedure is explained in the Figure 6-5, where the example saliency maps S_{I_0} and S_{I_R} images are given by the DRFI method [67].

Once again, the SEE-Post works with different saliency methods, but DRFI is chosen to better observe the improvements generated by the SEE method. In Figure 6-5, we observe how the intensity of the person is significantly enhanced compared to the background. The improvement is shown in the precision/recall curve of Figure 6-5 where the $S_{I_{0R}}$ curve has better or equal precision than S_I at every point. The curve is steady and somewhat similar to the one from SEE-Post in Figure 6-4, although the contrast between the salient person and the background is higher with the SEE-Pre than with the SEE-Post.

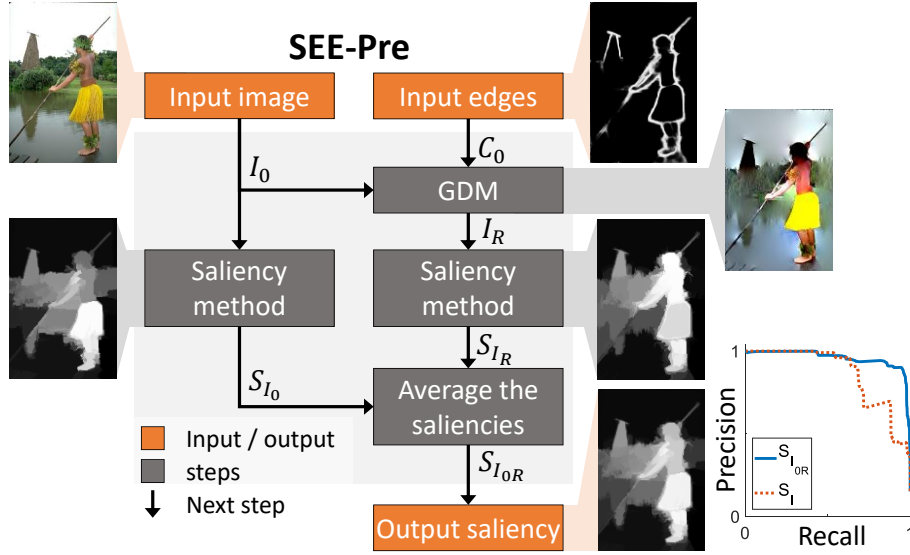


Figure 6-5 : Diagram of the SEE-Pre method, which allows preprocessing an image based on the salient edge detection for a more accurate saliency map. The images are examples of the results at each step based on the DRFI method with a precision/recall curve for the selected example.

For the GDM step of Figure 6-5, we can first ignore the preparation steps for the input edges $P_{C_0 \rightarrow 1}$ and for the input edges $P_{I_0 \rightarrow 1}$, meaning that $C_1^{\text{Pre}} = C_0^{\text{Pre}}$ and $I_1^{\text{Pre}} = I_0^{\text{Pre}}$.

Now we need to define the functions M_E^{Pre} and M_θ^{Pre} from equation (73) to determine how the gradient merging is done. The merger function for the norm M_E^{Pre} is defined in equation (83), which is similar to equation (81), but averaged with the field $|E|$ to preserve part of the texture and ensure the closure of the regions.

$$|E_p^{\text{Pre}}| = M_E^{\text{Pre}}(C_1, E) = \frac{\sqrt{|C_1| \circ |E|} + |E|}{2} \quad (83)$$

Then we define the merger function for the orientation M_θ^{Pre} simply uses the same angle as the gradient in order to preserve the texture and coloring of the image, as defined in equation (84).

$$\theta_p^{\text{Pre}} = M_\theta^{\text{Pre}}(C_1, E) = \theta_E \quad (84)$$

In summary, the general steps for SEE-Pre method are presented in Figure 6-5, with the specific flexible GDM steps presented in the list below.

- Contour preparation $P_{C_0 \rightarrow 1}$ and image preparation $P_{I_0 \rightarrow 1}$ steps are ignored.

- Norm merging function M_E^{Pre} defined in equation (83).
- Orientation merging function M_θ^{Pre} returns simply θ_E according to (84).

6.2.4.3 Contrast enhancement

Due to the use of inclusion probabilities and multiple averaging, it is preferable to enhance the contrast of the resulting image as suggested by our previous work on improving the probability of inclusion [16]. Hence, we compute the contrast-enhanced saliency S_C by using the smooth-step provided in equation (85) and based on Hermite polynomial [16,111], where S is any saliency map and S_C is the contrast enhanced saliency map. This polynomial enhances the value of any $S > 0.5$ and reduce the value of any $S < 0.5$. For the example of Figure 6-1, we use $K = 4$, with more detailed explanations to follow.

$$S_C = S^{K+1} \sum_{k=0}^K \binom{K+k}{k} \binom{2K+1}{K-k} (-S)^k \quad (85)$$

In the smooth-step equation (85), K is a parameter representing the intensity of the contrast enhancement, with $K = 0$ representing no contrast enhancement and $K \rightarrow \infty$ representing a step function that sets to 0 any value of $S < 0.5$ and to 1 any value of $S > 0.5$. In our current work, we use $K = 4$ since it helps improve the mean absolute error defined in equation (118), without any noticeable impact on the other parameters. Examples of the polynomial (85) for $K = \{2, 4\}$ are presented in equation (86). Furthermore, Figure 6-6 allows to better visualize the effect of the smooth-step function.

$$\begin{aligned} S_{C(K=2)} &= 6 S^5 - 15 S^4 + 10 S^3 \\ S_{C(K=4)} &= 70 S^9 - 315 S^8 + 540 S^7 - 420 S^6 + 126 S^5 \end{aligned} \quad (86)$$

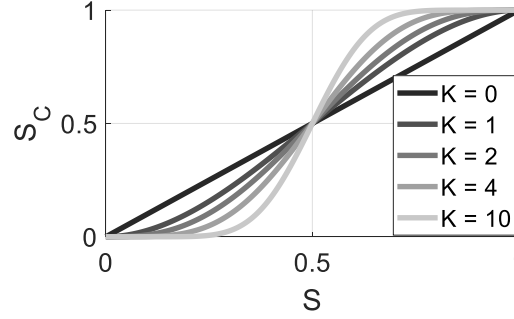


Figure 6-6 : Smooth-step function (85) with different values of K .

In Figure 6-1 representing the whole SEE method, the saliency $S_{I_{ORC}}$ is the application of the equation (86) on the saliency $S_{I_{OR}}$ resulting from the combination of SEE-Pre and SEE-Post.

6.2.5 Evaluation datasets and metrics

To properly evaluate our proposed SEE algorithm, we need to use standard datasets and metrics. For the datasets, we use the MSRA10K [122] for training purposes, which is an extension of the previous MSRA-B [67] dataset. The MSRA10K is used for training since it has the largest number of images (10,000). It is also one of the easiest in terms of performance which makes it is harder to discriminate between different algorithms, and it is the most used for training purposes [26].

For evaluation purposes, we use the following 3 datasets: ECSSD with 1000 images [68,124], PASCAL-S [125] with 850 more complex images and DUT-OMRON with the most complex 5168 images [126]. These datasets are used since they are among the standard in the literature for test evaluation purposes and they are used for the BGOF method [78].

For the comparison with other techniques, the parameters that are evaluated are the precision P , the recall or true positives R and the false positives 1R [69,127]. Those parameters are evaluated for 256 levels of thresholds on the saliency map S , which allows to plot the precision-recall PR curve. At each threshold level, a binary mask M is generated and compared to the binary ground-truth G . From the PR curve, one can evaluate the average \overline{PR} , the F-measure F_m and the maximum precision P_{\max} . All those parameters are defined in equations (112)-(116), where $\beta = 0.3$ is a constant that allows to add more weight to the precision, 1 is the logical NOT operator, \cap is the logical AND operator and \sum is the sum over every pixel [69,127].

$$P = \frac{\sum M \cap G}{\sum M} \quad (87)$$

$$R = \frac{\sum M \cap G}{\sum G}, \quad \text{!}R = \frac{\sum M \cap \text{!}G}{\sum \text{!}G} \quad (88)$$

$$P_{max} = \max(P) \quad (89)$$

$$F_m = \max\left(\frac{(1 + \beta^2)(P \text{ } R)}{\beta^2 P \text{ } R}\right) \quad (90)$$

$$\overline{PR} = \int P \text{ } dR \quad (91)$$

Other important information is the area under the curve (AUC) of the true-false-positive curve, and the mean absolute error (MAE) given respectively in equations (117) and (118), where S is the saliency map normalized to $[0, 1]$ and N the total number of pixels.

$$AUC = \int R \text{ } d\text{!}R \quad (92)$$

$$MAE = \frac{1}{N} \sum |S - G| \quad (93)$$

From all those parameters, the most used in the literature are the precision-recall PR curve, the F-measure F_m and the mean absolute error MAE . Hence, those parameters will be used to be compared with other methods from the literature. Additionally, we use the maximum precision P_{max} , the mean precision-recall \overline{PR} and the area under curve AUC to show that our approach improves many different measures.

6.3 Results

The complete SEE method was presented within the last section, but with only a single image example for the results. This section will show how the SEE methods on different images of the datasets ECSSD [68,124], PASCAL-S [125] and DUT-OMRON [126]. Different image examples will be provided, along with a benchmarking of the computation time.

6.3.1 Results on different images

The saliency methods on which the SEE is tested are DSS [11,26], DCL+ [76] (which combines DCL with denseCRF [79]), DRFI [67], RBD [74] and MC [128]. As we observe on the different examples in Figure 6-7, the SEE method significantly reduces the background values while enhancing the foreground values. Also, it partially fills some missing regions of the salient objects. Since a good saliency map is one where all the foreground values are higher than the background values, then we understand visually how our SEE method improves the best tested algorithms such as DCL [76] and DSS [26].



Figure 6-7 : Comparison of the results from 5 different SoA methods (MC, RBD, DRFI, DCL, and DSS) and their improvement using our SEE algorithm highlighted in green. The examples are chosen as some of the most difficult images in the datasets.

6.3.2 Computation time

Since our proposed SEE approach combines a SoA salient object detection with a SoA edge detection, then it is necessarily slower than the salient object detection alone. However, since many methods use denseCRF to improve their results [26,76], our method might perform faster by removing this layer. For our benchmarks, we use a graphics processing unit (GPU) Nvidia® GTX 1080 Ti and a central processing unit (CPU) Intel® i7-6700K.

For the GDM part, which is fundamental to the SEE algorithm, the computation time is around 1.5ms per RGB channel of size 400×400 on the GPU [17], which means that all the required GDM take around 6ms per image. This is negligible in the full computation time.

For the edge detection time, we use the RCF method, which takes around 100ms to compute a multiscale result [25]. For the salient object detection, the DCL method takes around 1000ms (due to the segmentation), while the DSS method takes around 80ms [26]. Both methods require an additional 400ms for using denseCRF to improve their saliency map [26]. Our method allows to remove the denseCRF but needs to compute the saliency twice. Hence, there is a computation time improvement for DSS, but not for DCL. The total computation time of the DSS-SEE method is less than 300ms while it is around 500ms for DSS-denseCRF. The total computation time of the DCL-SEE method is around 2000ms while it is around 1400ms for DCL-denseCRF.

6.4 Literature comparison and discussion

This section will perform a thorough benchmarking of the SEE-Post and the SEE methods on the datasets ECSSD [68,124], PASCAL-S [125] and DUT-OMRON [126]. The benchmarking includes measurements for the improvements over the saliency methods DSS [11,26], DCL+ [76], which combines DCL with denseCRF [79], MDF [87], DRFI [67], RBD [74] and DSR [71]. It also includes a comparison to state-of-the-art (SoA) methods for saliency maps improvement SO [74], denseCRF [79] and the most performant BGOF [78].

6.4.1 Improvement of the saliency maps

By using 3 different datasets and 7 SoA saliency methods, the current section shows that our proposed SEE approach allows to significantly improve the saliency results of many SoA algorithms, including the most recent CNN-based methods such as MDF, DSS, and DCL+.

In Figure 6-8, we can observe that both the SEE-Post and the SEE methods allow improving all 5 metrics defined in section 6.2.5 on the ECSSD (E), PASCAL-S (P-S) and DUT-OMRON (D-O) datasets. In fact, the only metrics where SEE reduces the performance is the P_{\max} for the DSS method.

The improvement is high enough that some less performant methods can outperform methods that are significantly better. Hence, we observe that MC with SEE outperforms DRFI on many measures and that DRFI with SEE outperforms MDF on many measures.

Also, we observe in Figure 6-8 that some methods receive a higher boost of performance than similar performing methods since they are naturally more adapted to the SEE method. For example, DSR is less improved than similar performing methods such as RBD, since the gradient of the RBD method merges better with the salient edge detection. Also, we note that the best regular method is DSS, but the best method is DCL+ when using the SEE algorithm. Again, this is because DCL+ merges better in the gradient domain, which gives it a bigger boost, especially for F_m , \overline{PR} and P_{\max} .

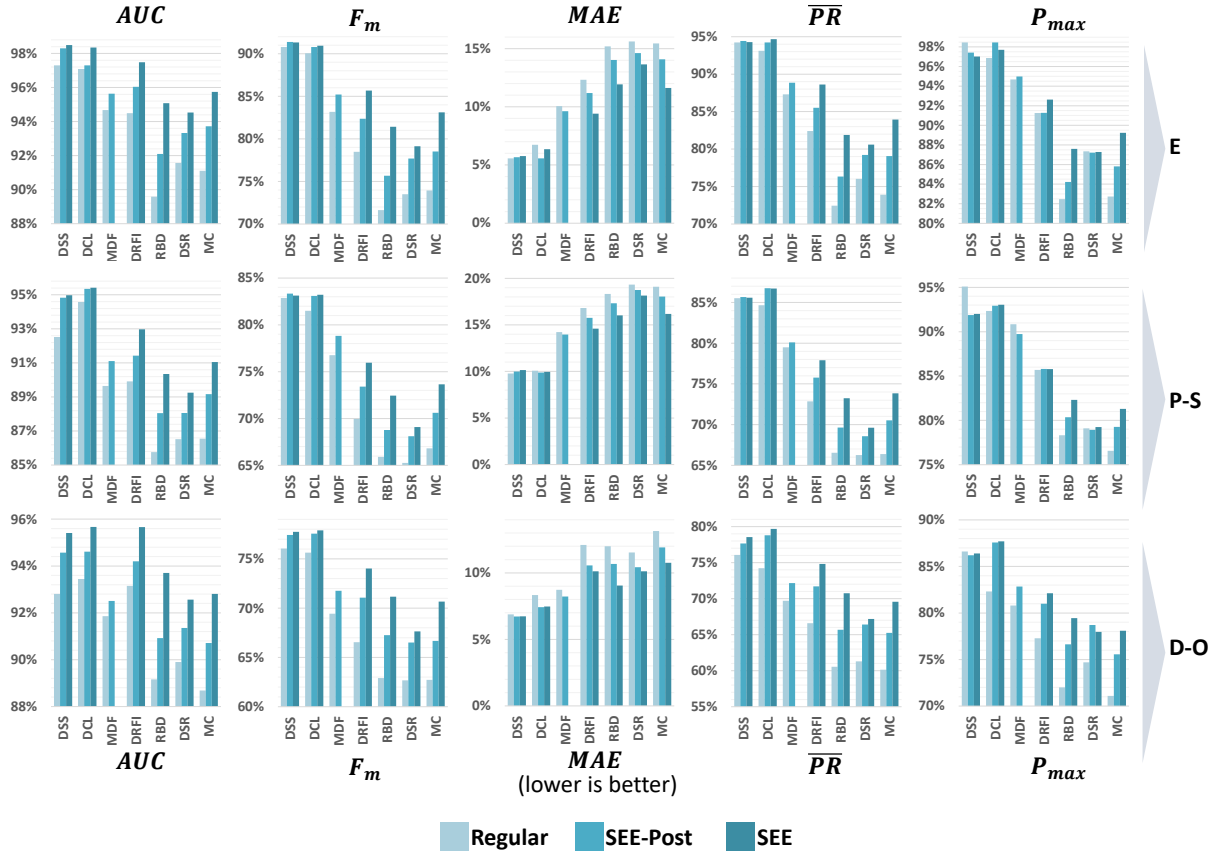


Figure 6-8 : Comparison between the regular saliency methods, the enhanced saliency using our SEE-Post algorithm and the enhanced saliency using our SEE algorithm. A higher percentage is better for AUC , F_m , \overline{PR} and P_{max} , but a lower percentage is better for MAE . The 3 datasets used are ECSSD (E), PASCAL-S (P-S) and DUT-OMRON (D-O).

For the same methods, we can observe the precision-recall PR curves on Figure 6-9 on the same 3 datasets. We observe that for any non-CNN based saliency method, the improvement of the SEE method is very high at every point of the PR curves. However, for the CNN-based methods, the regular method sometimes outperforms the SEE improvement at low recall ($R < 0.5$), but never at high recall ($R > 0.5$). Also, the PR curve is a lot flatter with the SEE method, meaning that the precision is almost constant for every recall that is not too high ($R < 0.8$). This is reflected in the improvement of the AUC and the F_m parameters in Figure 6-8. The flat curve means that the saliency is more robust with the SEE method and will lead to easier thresholding and more robust thresholding. Furthermore, we observe again that the DCL with SEE outperforms the DSS with SEE, although DSS outperforms DCL.

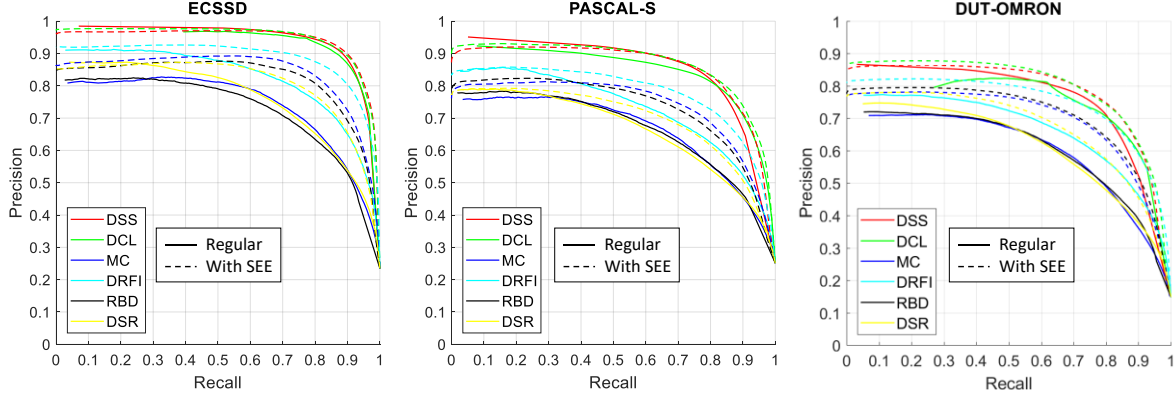


Figure 6-9 : Precision-recall curves comparing 6 different methods (DSR, RBD, DRFI, MC, DCL, and DSS) with and without our SEE method on 3 different datasets.

6.4.2 Comparison of saliency improvement methods

The previous section shows that our SEE method works well for its purpose of improving the saliency, and this section pursues by showing that it performs far better than any other algorithm with the same goal.

In fact, the improvement of SEE over F_m is on average 6.6 times better than the nearest competing algorithm BGOF [78] on ECSSD and 3.4 times on DUT-OMRON, as observed on Figure 6-10. Also, other methods such as SO [74] and denseCRF [79] are even further behind. Since F_m is the most universally used measure [26,62,67,69,76,127], this is an important achievement for the SEE method. Furthermore, the performance is better for MAE , another widely used indicator [67,69,76]. However, MAE is not as important as F_m since it can easily be improved by enhancing the contrast of the saliency map, while F_m is non-trivial to improve and indicates a direct improvement in the PR curve.

In summary, we observe that the proposed SEE method and even the proposed SEE-Post are a vast improvement compared to any other algorithm present in the literature. Hence for computation time reasons, one can choose to not use the SEE-Pre method, especially for the high-performance methods such as DCL and DSS where most of the improvements happen with the SEE-Post part of our approach.

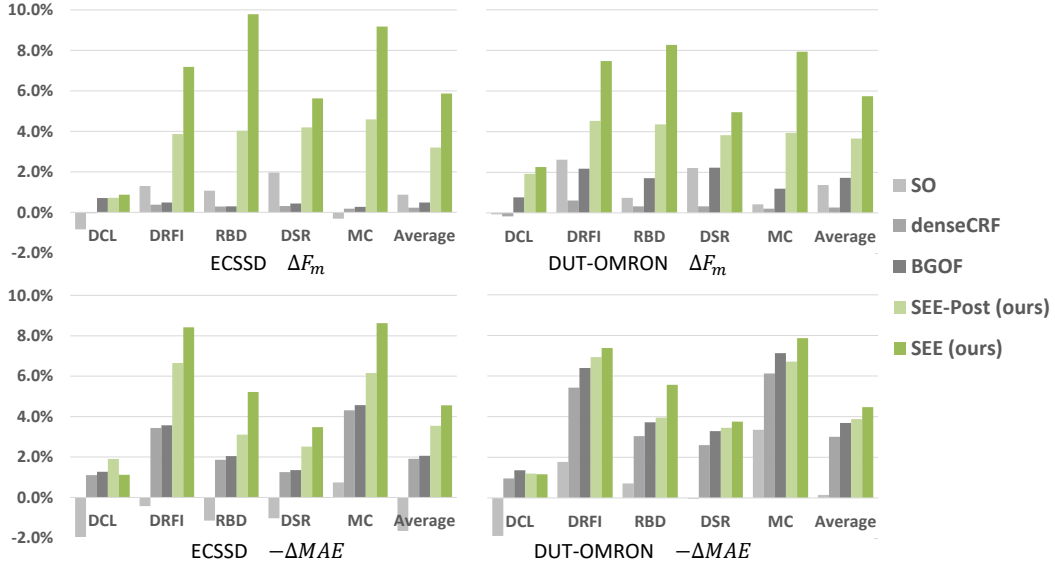


Figure 6-10 : Comparison of the improvement over F_m and MAE for different SoA saliency improvement methods.

6.5 Conclusion

The objective of this paper was to develop a novel method of merging the edges with the saliency maps to improve the performance of the salient object detection. It is the first work that allows combining the best advances in edge detection with the best advances in salient object detection. As seen in Figure 6-7, it works intuitively by reducing the values of the saliency map outside salient edges and enhancing it inside them, which is similar to how a human will perceive a salient object by its inclusion within boundaries. When compared to other methods of improving saliency maps with Figure 6-10, the SEE method shows an average improvement of the F-measure F_m 3.4 times more than the BGOF on the DUT-OMRON dataset and 6.6 times on the ECSSD dataset, and an improvement of the mean absolute error significantly better than all its competitors. We also showed how the SEE method improves by a high margin the precision-recall curve and some other measures such as the AUC , \overline{PR} and P_{\max} .

We believe that the proposed SEE method will have an important impact for the binary problems of computer vision since SEE is the first method that allows merging edge detection methods with saliency detection methods for improved results. A limitation of the method is that it needs 2 different neural networks trained separately which requires to optimize the parameters of both networks simultaneously and which increases the computation time. Future work can focus on

integrating the SEE method directly inside a neural network so that a single network is used, and all the parameters are optimized during the training process.

Acknowledgment

The authors are grateful to NSERC, through the discovery grant program RGPIN-2014-06289, and FRQNT/INTER for their financial support.

CHAPTER 7 ARTICLE 5: DEEP GREEN FUNCTION CONVOLUTION FOR IMPROVING SALIENCY IN CONVOLUTIONAL NEURAL NETWORKS

Dominique Beaini^{a1}, Sofiane Achiche^a, Alexandre Duperré^a, Maxime Raison^a

^aPolytechnique Montreal, 2900 Edouard Montpetit Blvd, Montreal, H3T 1J4, Canada

Submitted to Springer The Visual Computer, March 2019

Abstract

Current saliency methods require to learn large scale regional features using small convolutional kernels, which is not possible with a simple feed-forward network. Some methods solve this problem by using segmentation into superpixels while others downscale the image through the network and rescale it back to its original size. The objective of this paper is to show that saliency convolutional neural networks (CNN) can be improved by using a Green's function convolution (GFC) to extrapolate edges features into salient regions. The GFC acts as a gradient integrator, allowing to produce saliency features from thin edge-like features directly inside the CNN. Hence, we propose the gradient integration and sum (GIS) layer that combines the edges features with the saliency features. Using the HED and DSS architecture, we demonstrated that adding a GIS layer near the network's output allows to reduce the sensitivity to the parameter initialization, to reduce the overfitting and to improve the repeatability of the training. By simply adding a GIS layer to the state-of-the-art DSS model, there is an absolute increase of 1.6% for the F-measure on the DUT-OMRON dataset, with only 10ms of additional computation time. The GIS layer further allows the network to perform significantly better in the case of highly noisy images or low-brightness images. In fact, we observed an F-measure improvement of **5.2%** when noise was added to the dataset and **2.8%** when the brightness was reduced. Since the GIS layer is model agnostic, it can be implemented into different fully convolutional networks. Further, we showed that it outperforms the denseCRF post-processing method and is 40 times faster. A major contribution of the current work is the first implementation of Green's function convolution inside a neural network, which

¹ * Corresponding author. Tel.: +1-514-340-4711 # 3345

² E-mail address: dbeaini.phd@outlook.com

allows the network, via very minor architectural changes and no additional parameters, to operate in the feature domain and in the gradient domain at the same time.

Keywords: Salient object detection, Green's function convolution, Gradient integration sum, Saliency improvement, Deep learning

7.1 Introduction

Since the year 2015, the convolutional neural networks (CNN) rose quickly to become the best machine learning technique used to solve the binary problems of computer vision such as edge detection [25,66], skeleton extraction [11] and salient object detection [26,76]. In fact, recent algorithms perform near human-level [25].

At first, saliency methods were based on pre-programmed features such as clustering and density [69–71], concavity [72], contrast filtering [73], background detection [74], etc. Although they showed some success with simple images, they did not perform well on more complex dataset images [126]. The method DRFI [67] was the first to use machine learning, but it was soon outpaced by the arrival of CNN-based algorithms with methods such as MDF [87], DCL [76] and DSS [11,26]. The deeply supervised saliency (DSS) method was successful due to the efficient down-scaling and up-scaling of saliency maps.

An important problem with current salient object detection solutions is that they focus on finding the salient regions with little consideration to the fact that they are often bounded within edges. To overcome this limitation, some methods such as MDF [87] and DCL [76] use a pre-segmentation of the image. Furthermore, most methods fine-tune their results using saliency enhancement methods such as the denseCRF [79] algorithm during the testing phase, which uses segmentation to clean the saliency maps and make it more accurate to the boundaries.

Different methods of enhancing the saliency maps are proposed in the literature. WCtr [74] proposes to improve the saliency maps using background detection. However, BGOF [78] showed that most saliency improvement algorithms based on segmentation and background detection do not work on recent networks since CNN are better at detecting the background and segmentation than traditional methods. In contrast, algorithms such as denseCRF [79] and BGOF [78] optimize the saliency map density via energy minimization. The DeepSets method [129] is very similar since it uses super-pixels to enhance the saliency maps on the boundaries and increase the density.

Alternatively, the RACDNN method [130] proposes to use a recurrent attention mechanism to recursively enhance each region of the saliency map. Although RACDNN trains with the network, the attention mechanism is outside it and significantly increases the architectural complexity by adding recurrent layers [130]. As explained in later paragraphs, the proposed method differs from the literature since it adds a layer directly inside the network, thus directly improving the capacity of the network to generate saliency maps.

The objective of this paper is to show that a saliency CNN can be improved using a Green's function convolution (GFC), which allows integrating edge-like features into salient features. Hence, we propose the gradient integration and sum (GIS) layer, which integrates the gradient domain features and adds them to the special domain features. By doing so, the GIS layer creates a smooth and continuous region between the high gradient boundaries, thus enhancing the saliency map inside boundaries and reducing it outside the boundaries. Hence, for the proposed method, the network directly trains the parameters used for the saliency improvement, in contrast with other methods which act outside of the main network.

It is to note that the saliency improvement occurs in the uniform regions inside boundaries, not at the boundaries themselves. This is because the GFC extrapolates the edges into regions of smooth probabilities within the image, as demonstrated by Beaini et al. [16].

The denseCRF [79], BGOF [78], DeepSets [129] and RACDNN [130] are methods that post-process the saliency maps outside of the neural networks, aiming to improve the boundaries and to increase the density of the maps. This is in contrast with the proposed method consisting of adding a GIS layer directly inside the network, thus allowing the network to train its inputs. Furthermore, GIS does not aim at improving the density or the boundaries but consists of allowing the network to combine features from the saliency-domain and the gradient-domain.

The GIS is first proposed in this paper, although similar concepts of gradient-domain merging were previously both proposed by Beaini et al [17]. Previous work using Green's function convolution (GFC) for Poisson image editing, contrast enhancements and paint-like effects [17,113,115,119]. To our knowledge, they are never implemented inside neural networks. However, the GFC method was demonstrated to solve 100 Laplacians in 1ms using machine learning libraries such as Pytorch and Tensorflow [17].

The idea of merging edges with saliency inside the network comes from the fact that edge detectors are fast to learn by a CNN since they require gradient-like [63] or Gabor-like kernels [9,49]. Thus we propose to create a network that computes saliency and edge-like features at the same time, then merges them using a GIS layer. Further, to better understand the importance of the edges in the saliency detection, we visualize the inputs and outputs of the GIS layer at different scales. The current paper demonstrates that the proposed GIS layer improves salient object detection for different network architectures, resulting in better accuracy, less overfitting and lower sensitivity to the network initialization.

In our work, we propose to use the GIS layer on the HED [66] and DSS [11,26] architectures by adding our layers to the end of each side-layer of the original networks, without any other architectural changes. Both HED [66] and DSS [11] are known for their edge detection performance, but only DSS [26] performs well for salient object detection. However, our work shows that the GIS layer improves the HED network by a high margin, thus allowing it to outperform saliency-focused networks. We will refer to the modified models as HED-GIS and DSS-GIS.

In summary, our contributions are that we are the first to implement a GFC-based layer inside a CNN and that such layer allows to significantly improve salient object detection by extrapolating edges into smooth saliency maps.

7.2 Methodology

The full implementation of DSS-GIS is done with Python using the TensorFlow library. The current section will explain how the GIS work, what are the HED and DSS architectures and how we modify them to create the proposed HED-GIS and DSS-GIS.

7.2.1 Gradient integration and sum (GIS)

The GIS method is first proposed in the current work but is inspired by the field of *gradient-domain image editing*, which mainly focuses on applying editing filters to images [58,112]. We mainly base our work on the GFC method proposed by Beaini et al. [17] which allows integrating any vector field with minimal error and low computation time (100 images in 1ms). For the current paper, we are interested in the ability of the GIS to combine edge-like features (object boundaries)

with region-like features (saliency maps). Those features cannot be combined by standard operations such as additions, multiplications or small-kernel convolutions since most edge pixels do not intersect saliency pixels. Hence, before summing the features, an integration step is required to transform the edge-like features into region-like features. The following subsections will explain how the GIS integrates the gradient and merges it with the standard features.

7.2.1.1 Green's function convolution (GFC)

The current subsection focuses on the gradient-integration step and is based on work by Beaini et al. [17].

Let us denote \mathbf{E} as a vector field of features made of the horizontal E_x and vertical E_y components. The vector field \mathbf{E} cannot be integrated directly since it is not necessarily a conservative field, meaning that it does not have a solution.

Hence, we use Green's function based solver proposed by Beaini et al. [17]. We first need to compute the Laplacian L_p , then to solve it using a Green's function convolution (GFC), as described in this section.

The Laplacian L_p is computed using equation (94), where $E_{x,y}$ are the x and y components of the field \mathbf{E} and $K_{E \rightarrow L}$ is the convolutional kernel that represents this operation.

$$L_p = \frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} = K_{E \rightarrow L} * \mathbf{E} \quad (94)$$

Now that the Laplacian is computed, we need to compute the Green's function that solves it. The Green's function is defined as a function that solves a given differential equation with a convolution [33]. In our case, the differential equation is the numerical Laplacian given by the convolution in equation (95). In this equation, I is any image, L_I is its Laplacian and K_{∇^2} is the Laplacian kernel.

$$L_I = I * K_{\nabla^2}, K_{\nabla^2} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (95)$$

If we denote V_{mono} as being the numerical Green's function that solves the Laplacian, then equation (96) shows that the convolution $K_{\nabla^2} * V_{mono}$ act as an identity. Since the convolution identity is the Dirac's delta δ [33], then equation (97) represents this relation.

$$I = I * K_{\nabla^2} * V_{mono} \quad (96)$$

$$K_{\nabla^2} * V_{mono} = \delta \quad (97)$$

Then we define the convolution theorem [33] in equation (98) where A, B are any function, \mathcal{F} is the Fourier transform, \mathcal{F}^{-1} is the inverse Fourier transform and \circ is the element-wise product.

Using equation (98) it becomes possible to solve equation (97) for $\mathcal{F}(V_{mono})$, as given by equation (99). The notation $V_{mono}^{\mathcal{F}}$ represents the Green's function in the Fourier domain.

$$A * B = \mathcal{F}^{-1}(\mathcal{F}(A) \circ \mathcal{F}(B)) \quad (98)$$

$$V_{mono}^{\mathcal{F}} = \mathcal{F}(V_{mono}) = \frac{\mathcal{F}(\delta)}{\mathcal{F}(K_{\nabla^2})} \quad (99)$$

Equation (99) gives a solution for the Green's function V_{mono} . However, to be applied on an image as given by equation (96), V_{mono} must be the same size as the image.

Hence, we define $\check{\delta}$ as the padded numerical Dirac's delta and \check{K}_{∇^2} as the padded numerical Laplacian kernel in equation (100), where L_I is the Laplacian to solve [17]. Then, the Green function in the Fourier domain $\check{V}_{mono}^{\mathcal{F}}$ is given by (101) [17].

$$\check{K}_{\nabla^2} \equiv \begin{bmatrix} 0 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & & \\ 0 & -1 & 0 & & \\ \vdots & & & \ddots & \\ 0 & & & & 0 \end{bmatrix} \quad \check{\delta} \equiv \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & \\ 0 & 0 & 0 & & \\ \vdots & & & \ddots & \\ 0 & & & & 0 \end{bmatrix} \quad (100)$$

$\underbrace{\hspace{10em}}_{size(L_I)} \quad \underbrace{\hspace{10em}}_{size(L_I)}$

$$\check{V}_{mono}^{\mathcal{F}} = \frac{\mathcal{F}(\check{\delta})}{\mathcal{F}(\check{K}_{\nabla^2})} \quad (101)$$

In equation (101), $\check{V}_{mono}^{\mathcal{F}}$ is the Green's function that allows to solve any Laplacian by a convolution [17,33]. The convolution is computed using the Fourier domain as defined in equation (102) since Fourier transforms are faster for large convolutions and are implemented on a graphical processing unit (GPU) in multiple machine vision libraries such as OpenCV [35], MATLAB® [36] and

Tensorflow. In equation (102), \mathcal{R} is the real part of a complex number, c is an integration constant and I_R is the resulting image. In practice, a 4-pixels padding of value 0 is added all around L_p to avoid discontinuities in the numerical Laplacian [17]. The constant c is equal to the values in the padded part of I_R .

$$I_R = \mathcal{R} \left(\mathcal{F}^{-1} \left(\mathcal{F}(L_p) \circ \check{V}_{mono}^{\mathcal{F}} \right) \right) - c \quad (102)$$

More details about the mathematical foundation of the Laplacian solver, as well as empirical demonstrations and pseudo-codes are provided in a previous work by Beaini et al. [17].

7.2.1.2 Overview of the GIS layer

To better understand the GIS layer, a graph is provided in Figure 7-1. We observe that GIS has n output channels for $3n$ input channels. The 3 input groups are S , E_x and E_y . S is considered in the spatial domain and is simply summed to the output. E is considered in the gradient domain and is integrated using GFC before being summed to S .

Note that a weighted sum is not required since the inputs are expected to be weighted by the CNN.

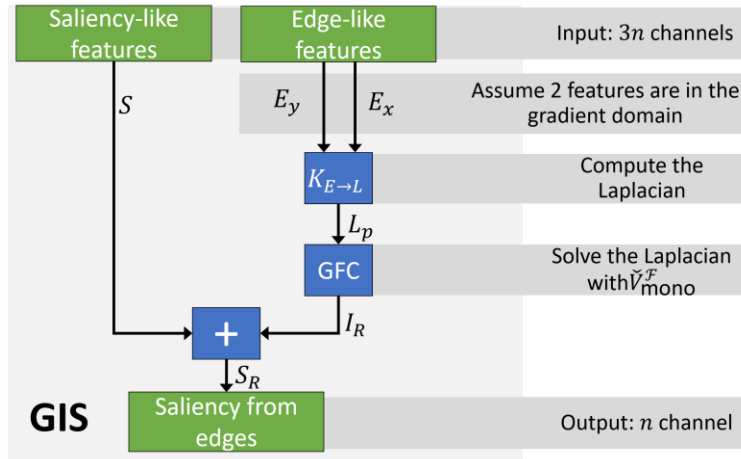


Figure 7-1 : Graph summary of the gradient integration and sum (GIS) layer, which outputs n channels from $3n$ inputs.

7.2.2 Implementing the models with the GIS layer

The proposed GIS layer can only be implemented on fully convolutional networks since they require that the network is able to output both saliency and edges at the same time. Hence, we use

the DSS model, which to the best of our best knowledge, is one of the most successful saliency model [11,26]. We also use the HED model to demonstrate that our approach is generalizable to more networks.

It is to note that GIS implements an integration of edge-like features from the gradient domain. However, the network is never forced, via an intermediate loss, to learn the gradient of the saliency map. The gradient of the saliency is naturally learned by the fact that the GFC gradient integrator is used, without requiring a saliency ground-truth. Hence, the edge-like features from Figure 7-1(E_x and E_y) are not necessarily the gradient of the saliency, although they are expected to be. In addition, the saliency-like features used by GIS in Figure 7-1 (S) is not forced to be similar to the saliency. It is simply expected to be similar to the saliency due to the merging operation. Hence, S cannot be used directly as an output of the network.

7.2.2.1 The HED and DSS models

The HED and DSS models are an architecture nested on top of a classification network, with deep side layers connected before every pooling [26]. They are presented in Figure 7-2, with the classification network being the pre-trained VGGnet-16 [10] presented in gray in Figure 7-2. They have a total of 6 side outputs with 3 layers each, with the first 2 layers being followed by a ReLU operation [26]. The side layers are presented in blue in Figure 7-2 with the parameters defined in a later section in Table 7.1. The only difference between the standard model and our model are that the 3rd layer of each side layer has only 1 output for HED/DSS instead of 3 outputs for HED/DSS-GIS. The weights of the side layers are initialized as a normalized uniform random distribution.

A closer view on the integration of the GIS layer within the network is presented in Figure 7-3, where we observe that each side layer sideX_3 produces 3 outputs, which are then split into the S , E_x and E_y inputs of the GIS layer.

All the parameters of the DSS-GIS are summarized in Table 7.1 and the architecture is summarized in Figure 7-2.

For the maximum performance of the DSS-GIS, the GIS layers are expected to have one saliency-like input in the spatial domain and 2 inputs in the gradient domain, since it is how the GIS layer was designed. This is indeed what is observed in Figure 7-4 where S distinguishes the regions and $E_{x,y}$ highlight the edges of the people sitting in the grass.

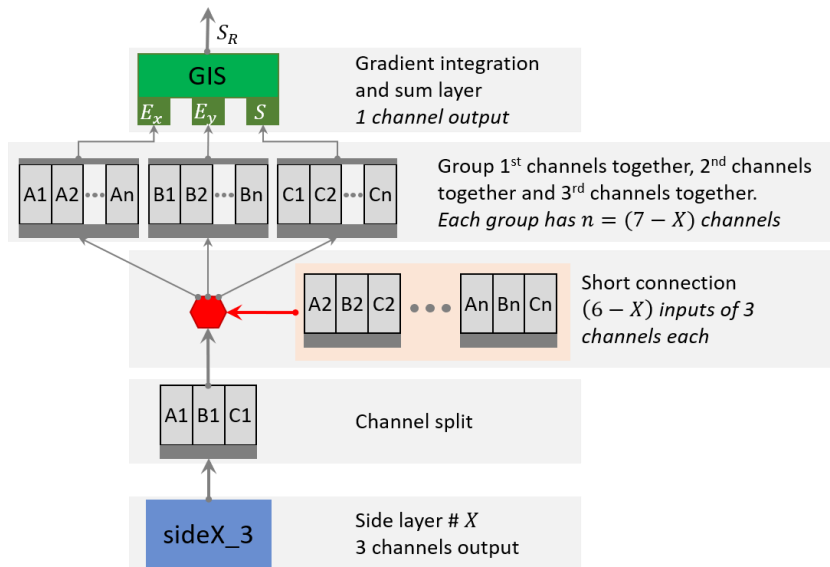


Figure 7-3 : Closer view on the integration of the GIS layer inside the DSS architecture. For the HED architecture, the GIS layer is placed directly after the channel split since there are no short connections.

Table 7.1 : Side layer information of HED and DSS architectures given by $(n, k \times k)$, where n is the number of output channels and $k \times k$ is the size of the kernel. “Layer” is the name of the layer from the VGGnet-16 whose output is connected to a side layer. “1”, “2” and “3” represent the 3 layers for each side output. “1” and “2” are followed by a ReLU operation. If a GIS layer is added, $n_o = 3$, otherwise $n_o = 1$.

No.	VGG layer	sideX_1	sideX_2	sideX_3
1	conv1_2	$128, 3 \times 3$	$128, 3 \times 3$	$n_o, 1 \times 1$
2	conv2_2	$128, 3 \times 3$	$128, 3 \times 3$	$n_o, 1 \times 1$
3	conv3_3	$256, 5 \times 5$	$256, 5 \times 5$	$n_o, 1 \times 1$
4	conv4_3	$256, 5 \times 5$	$256, 5 \times 5$	$n_o, 1 \times 1$
5	conv5_3	$512, 5 \times 5$	$512, 5 \times 5$	$n_o, 1 \times 1$
6	pool5	$512, 7 \times 7$	$512, 7 \times 7$	$n_o, 1 \times 1$

In Figure 7-4, We observe that S_R is mainly driven by the S input for the side layer #5 where the resolution is low, but it is mainly driven by the integration over $E_{x,y}$ for the side layers #3, 4 where the resolution is high. This is because the convolutional kernels are too small to detect regions for the high resolution layers. However, they are able to detect edges, which can be integrated into regions via the GIS layer.

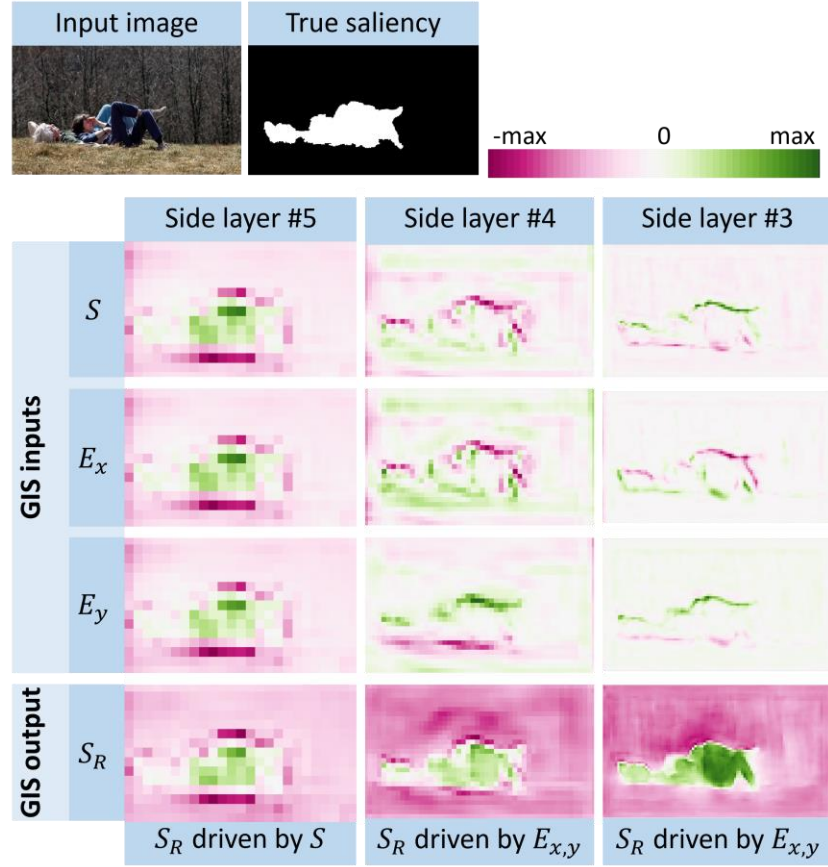


Figure 7-4 : Example of the inputs of the GIS layer coming from a fully trained HED-GIS network. S is expected to be in the saliency domain; E_x, E_y are expected to represent the 2 components of the gradient domain.

In summary, the GIS layers act in a similar way to an activation function at the deepest side-layers of the network since they perform a pre-defined operation on the input channels. However, as shown in Figure 7-1, GIS outputs a third of its input channels, and it performs a gradient integration operation with a summation. Hence, the GIS layer does not use any weight or intermediate loss.

7.2.2.3 Training procedure

An important modification from the DSS model is that the original code is in Caffe [26] but we recoded the entire architecture in Tensorflow to make use of its multi-platform capabilities, the integrated Fourier transforms, the improved convergence algorithms and the real-time validation curves.

For the parameters, the DSS model [26] proposes to use 10 images per mini-batch. It also uses a standard gradient descent with a learning rate of 10^{-8} for 20k iterations and 10^{-9} for an additional 4k iterations.

In contrast, we changed those parameters to 8 images per mini-batch, with an Adam optimizer [131] and a learning rate of $4 \cdot 10^{-5}$ for 30k iterations. We use an early stopping method to save the model with the highest F-measure (defined in section 7.3) on the validation set, then fine-tune the new model for 2000 iterations using a learning rate of $4 \cdot 10^{-6}$.

Those changes are made since our loss is computed as the average loss over the pixels instead of the sum, and because the Adam optimizer removes the need for changing the learning rate [131].

We also use the MSRA10K [122] for training purposes, which is an extension of the previous MSRA-B [67] used for DSS [26]. The MSRA10K is randomly split into 7000 training images, 2000 validation images and 1000 test images. Furthermore, the training images are duplicated using horizontal reflection leading to 14000 training images, as proposed by Hou et al. [26].

Finally, another change that is made to the model is that we use a zero-padding all-around the training images until they reach a resolution of 416×416 . Since every image of the MSRA10K dataset has a maximum resolution of 300×400 , this operation does not resize or crop the images. Furthermore, the computation of the loss, as well as the other measures presented in section 7.3, ignore all the padded pixels.

7.3 Evaluation datasets and metrics

To evaluate our proposed DSS-GIS algorithm, we need to use standard datasets and metrics. For the datasets, we use the MSRA10K [122] for training since it has the largest number of images (10,000). It is also the most used for training purposes [26]. We randomly split the MSRA10K into 7000 images for training, 2000 images for validation and 1000 images for testing.

For testing purposes, we use the following 3 datasets: ECSSD with 1000 images [68,124], PASCAL-S [125] with 850 more complex images and DUT-OMRON with the most complex 5168 images [126].

For the purpose of comparing the performances to other techniques, the parameters that are evaluated are the precision P , the recall or true positives R and the false positives 1R [69,127].

Those parameters are evaluated for 256 levels of thresholds on the saliency map S , which allows to plot the precision-recall PR curve. At each threshold level, a binary mask M is generated and compared to the binary ground-truth G . From the PR curve, one can evaluate the average \overline{PR} , the F-measure F_m and the maximum precision P_{\max} . All those parameters are defined in equations (112)-(116), where $\beta = 0.3$ is a constant that allows to add more weight to the precision, $^!$ is the logical NOT operator, \cap is the logical AND operator and \sum is the sum over every pixel [69,127].

$$P = \frac{\sum M \cap G}{\sum M} \quad (103)$$

$$R = \frac{\sum M \cap G}{\sum G}, \quad ^!R = \frac{\sum M \cap ^!G}{\sum ^!G} \quad (104)$$

$$P_{\max} = \max(P) \quad (105)$$

$$F_m = \max\left(\frac{(1 + \beta^2)(P R)}{\beta^2 P + R}\right) \quad (106)$$

$$\overline{PR} = \int P dR \quad (107)$$

Other important information is the area under the curve (AUC) of the true-false-positive curve (117), the mean absolute error (MAE) (118), the root mean square error (RMSE) (110) and the cross-entropy (CE) (111) [9]. In those equations, S is the saliency map normalized to $[0, 1]$, N the total number of pixels and G is the ground-truth with binary value 0 or 1.

$$AUC = \int R d^!R \quad (108)$$

$$MAE = \frac{1}{N} \sum |S - G| \quad (109)$$

$$RMSE = \sqrt{\frac{1}{N} \sum |S - G|^2} \quad (110)$$

$$CE = \frac{-1}{N} \sum G \log S + (1 - G) \log(1 - S) \quad (111)$$

From all those parameters, the most used in the literature are the precision-recall PR curve, the F-measure F_m and the area under the curve AUC [70,71,76,87,127,132]. These metrics are used since they represent better the effects of thresholds on the saliency maps and they cannot be improved with simple methods such as contrast enhancement. Hence, those parameters will be used to be compared with other methods from the literature. Additionally, we use the mean precision-recall \overline{PR} and the mean absolute error MAE to show that our approach improves many different measures.

It is worth noting that the cross-entropy CE is used as the loss function for the training of the DSS-GIS model.

7.4 Results

This section presents the saliency map results and a comparison of the validation curves. The results show that DSS-GIS has better saliency maps than DSS, trains faster, is less prone to overfit and has higher accuracy.

7.4.1 Saliency maps

The improved performance of our HED/DSS-GIS approach compared to the standard HED/DSS can be observed on some test images from the ECSSD, PASCAL-S and DUT-OMRON images in Figure 7-5. In this image, we see that the GIS layer improves the results when there is a bright contrast, a complex background, a camouflaged animal or small salient objects.

The improvements are very notable when looking at how much the HED-GIS outperforms HED by providing smoother and more accurate saliency maps in all examples. This is because the HED model lacks the upward scale introduced by the DSS short connections required for a good saliency prediction. However, since HED is a good edge predictor, the HED-GIS was still able to produce accurate saliency maps by extrapolating the edge-like features into the image space.

The improvements of DSS-GIS over DSS are subtle. We notice the improvement only for the most difficult examples presented in Figure 7-5. DSS-GIS is better at discriminating between the salient object and a given background, and better at finding a camouflaged animal or small objects. The reason is that, in these cases, relying on the globally strong edges is more important

than relying on the contrast and texture, since the background has as many colors and complexity as the foreground.

Furthermore, for the *incomplete saliency* category, the DSS-GIS allows finding more salient regions than DSS. The reason is that it fills up the missing saliency regions by ensuring that all the areas inside edges are included in the saliency map, as explained in previous work [16].

However, there are some failures cases of DSS-GIS presented in Figure 7-5. Those include images where the background has very strong defined edges, but the foreground does not. Also, DSS seems to perform better on transparent objects, since those objects are better detected by their glare than by their edges. It is to note that those failure cases do not apply for HED-GIS, which consistently outperforms HED.

7.4.2 Validation curves

The first difference that we notice when training the proposed DSS-GIS model is that its validation curves are far more similar, faster to train and less prone to overfitting than the DSS model. This can be observed in Figure 7-7 with 6 different training curves of the DSS in orange and 6 curves of the DSS-GIS in blue. Note that these curves are for a learning rate of 10^{-5} and different parameter initialization.

The DSS curves are **our** implementation of the model, meaning that it uses exactly the same code as the DSS-GIS, but without the GIS layer.

These curves are generated by computing the loss, the F-measure, the MAE and the RMSE at every 50 iterations, and by randomly selecting 200 images from the validation set. To speed up the computation, the F-measure uses 51 different thresholds instead of the standard 256 levels. Then, an exponential smoothing with a factor 0.9 is applied to all the curves to reduce the noise.

On Figure 7-7, we see that the DSS-GIS reaches a higher performance for the 4 different measures. Also, the DSS model has a big disparity between different validation curves, meaning that it is more sensible to the random initialization of the side layers and the different initialization algorithms. In fact, DSS does not always converge to its maximal performance if the initialization is not optimal. Finally, we can see that the CE loss of the DSS diverges at around 10k iterations but remains almost constant for DSS-GIS.

All those differences show that the DSS-GIS is easier and faster to train, less prone to overfit and leads to better results than the DSS model. Furthermore, the training is more robust to the random parameter initialization, leading to more similar training curves across different trainings. Since the only difference between the 2 models is a fast to compute GIS layer at each side output, we deduce that our proposed DSS-GIS outperforms the DSS model.

After optimizing the initialization parameters and the learning rate via cross-validation, we obtained the curves presented in Figure 7-6. In this figure, we observe that the optimal DSS converges as fast as DSS-GIS and has less overfit than the models in Figure 7-7. However, it still converges to a lower validation performance. We also observe that the HED-GIS model strongly outperforms the standard HED model.

In addition, the GIS layer seems to reduce the noise of the validation curves, meaning that the network converges more easily to the optimal solution. Hence, we conclude that the GIS layer allows to significantly improve the training process of the networks.

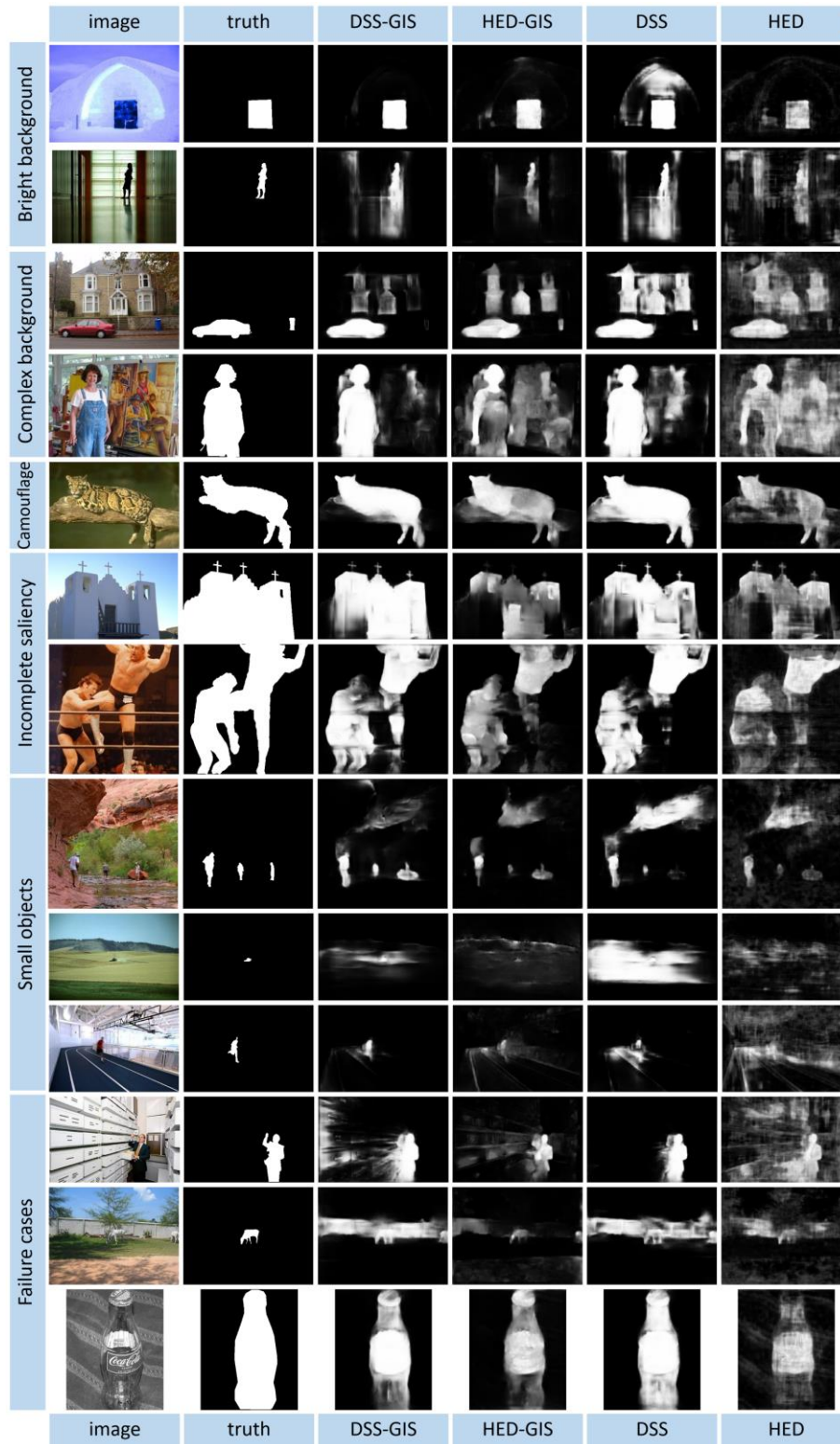


Figure 7-5 : Test results comparison between the DSS and HED methods with and without the proposed GIS layer.

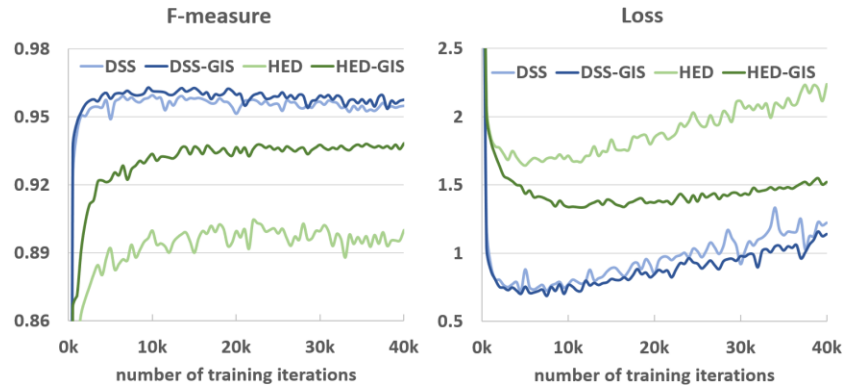


Figure 7-6 : Comparison of the validation curves of different models for the optimally found parameters. The validation performance is computed every 500 iterations on the full validation set.

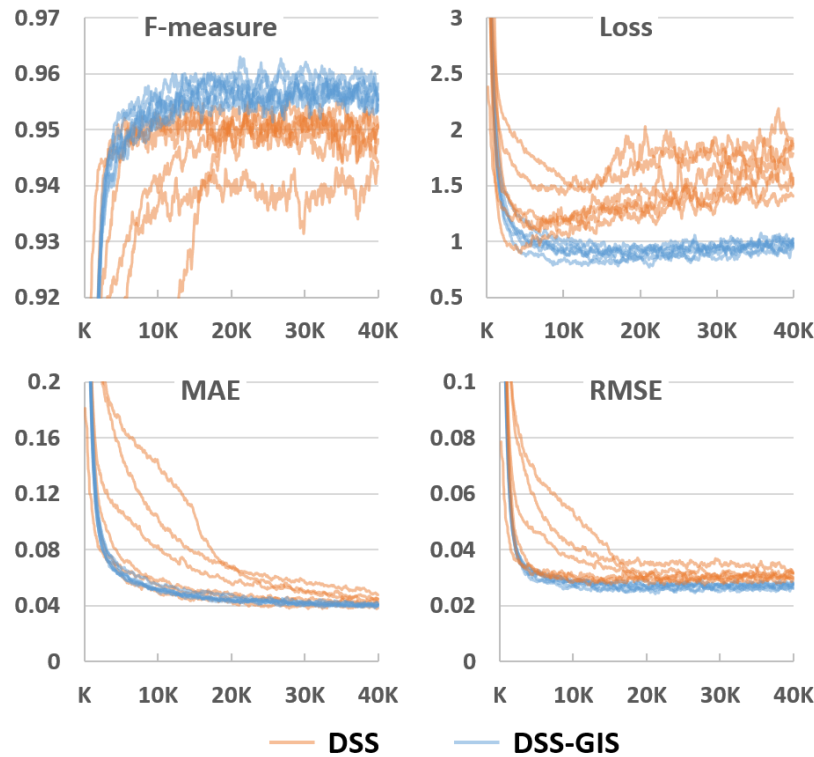


Figure 7-7 : Comparison of 6 validation curves in orange of our implementation of the DSS model, and 6 validation curves in blue of our DSS-GIS model. The curves are smoothed using exponential smoothing with a factor 0.9, and the x-axis represents the number of iterations.

7.5 Literature comparison and discussion

This section will compare our proposed HED/DSS-GIS models to multiple models in the literature using the metrics specified in section 7.3. It will show that the DSS-GIS model outperforms the other model on every dataset.

7.5.1 Training the DSS, DSS-GIS

For a fair comparison between the standard method and the same method with the GIS layer, we use exactly the same training procedure as the one defined previously in section 7.2.2.3. Furthermore, the random seed is the same for all models, meaning that the network initialization is identical across models and the training-validation-testing split is also identical.

7.5.2 Testing improvement of the GIS layer

In this section, we compare the results given by the DSS and HED models with and without the proposed GIS layer.

For the DSS method, our implementation performs better than the original implementation in their paper [132]. For a fair comparison, we thus use our implementation of DSS for all reported test results.

The compared results are shown in Table 7.2 for HED and Table 7.3 for DSS, where the * symbol means that a denseCRF layer is added. We observe that the proposed DSS-GIS is always better than DSS, and that the proposed HED-GIS is always better than HED.

Table 7.2 shows that GIS always outperforms denseCRF for all metrics for the HED model. Table 7.3 shows that GIS always outperforms denseCRF for the DSS model, except for MAE. However, the difference in MAE can be explained by the fact that denseCRF enhances the contrast.

Since the only differences between DSS-GIS and DSS are the added GIS layer from Figure 7-2, the testing results show that it is fully justified to use the GIS layer instead of denseCRF since most improvements are due to it. However, using both together usually yields the best results.

Table 7.2 : Comparison between the percentage testing results of HED and our HED-GIS. The * means that a denseCRF layer is added at the output. The best value in each category is highlighted in bold (ignoring GIS*). The values are in percentages.

Method	AUC	F_m	MAE	\overline{PR}	Dataset
HED-GIS*	+1.5	+3.6	-7.5	+3.2	ECSSD
HED*	+0.1	+1.2	-4.2	+0.5	
HED-GIS	+1.5	+3.1	-6.0	+2.9	
HED	96.2	85.2	16.4	88.9	
HED-GIS*	+3.5	+8.7	-9.0	+8.7	DUT-OMRON
HED*	+0.2	+2.0	-3.7	+1.0	
HED-GIS	+3.5	+7.6	-7.3	+8.3	
HED	90.7	66.6	17.8	66.2	
HED-GIS*	+1.3	+3.0	-7.2	+2.6	PASCAL-S
HED*	+0.1	+1.1	-4.1	+0.5	
HED-GIS	+1.3	+2.4	-5.3	+2.5	
HED	92.4	77.1	20.5	79.7	

Table 7.3 : Comparison between the testing results of DSS and our DSS-GIS. The * means that a denseCRF layer is added at the output. The best value in each category is highlighted in bold (ignoring GIS*). The values are in percentages.

Method	AUC	F_m	MAE	$\overline{(PR)}$	Dataset
DSS-GIS*	+0.1	+0.7	-0.7	+0.6	ECSSD
DSS*	-0.1	+0.3	-0.6	+0.0	
DSS-GIS	+0.3	+0.3	-0.1	+0.6	
DSS	98.3	91.5	6.1	95.5	
DSS-GIS*	+0.3	+1.8	-0.8	+2.1	DUT-OMRON
DSS*	-3.3	+0.3	-1.0	-2.6	
DSS-GIS	+0.5	+1.6	-0.2	+2.0	
DSS	95.8	76.6	8.0	78.8	
DSS-GIS*	+0.4	+0.4	-0.6	+0.6	PASCAL-S
DSS*	-0.2	+0.3	-0.7	0.0	
DSS-GIS	+0.6	+0.3	+0.2	+0.6	
DSS	94.8	83.3	11.0	86.6	

7.5.3 Literature benchmarking

In this section, we aim at demonstrating that the proposed DSS-GIS algorithm outperforms other state-of-the-art (SoA) algorithms.

In Table 7.4, we show that our DSS-GIS algorithm outperforms all the other tested methods in terms of F_m and AUC . The improvements are mostly notable on the DUT-OMRON dataset since it is the most difficult one, with the most complex backgrounds. We also note that HED performs badly compared to other algorithms, while HED-GIS is very close to the high performance DCL method.

We also observe on Figure 7-8 that the precision/recall curves and the true-positive/false-positive curves of the DSS-GIS consistently outperforms the other methods. Again, HED-GIS outperforms HED by a high margin.

Table 7.4 : Comparison of the proposed DSS-GIS and HED-GIS approaches (grey rows) with other saliency algorithms proposed in the literature. The best result of each column is highlighted in bold. The values are in percentages.

Dataset	ECSSD		DUT-OMRON		PASCAL-S	
Method	F_m	AUC	F_m	AUC	F_m	AUC
DSR [71]	73.5	91.6	62.7	89.9	65.3	86.5
RBD [74]	71.6	89.6	62.9	89.2	65.9	85.8
DRFI [67]	78.5	94.5	66.5	93.2	70.0	89.9
MDF [87]	83.2	94.7	69.4	91.9	76.8	89.7
DCL [76]	90.1	97.1	75.6	93.4	81.5	94.5
HED [66]	85.2	96.2	66.6	90.7	77.1	92.4
HED-GIS	88.3	97.7	74.1	94.3	79.5	93.7
DSS [26]	91.5	98.3	76.6	95.8	83.3	94.8
DSS-GIS	91.9	98.6	78.2	96.4	83.6	95.4

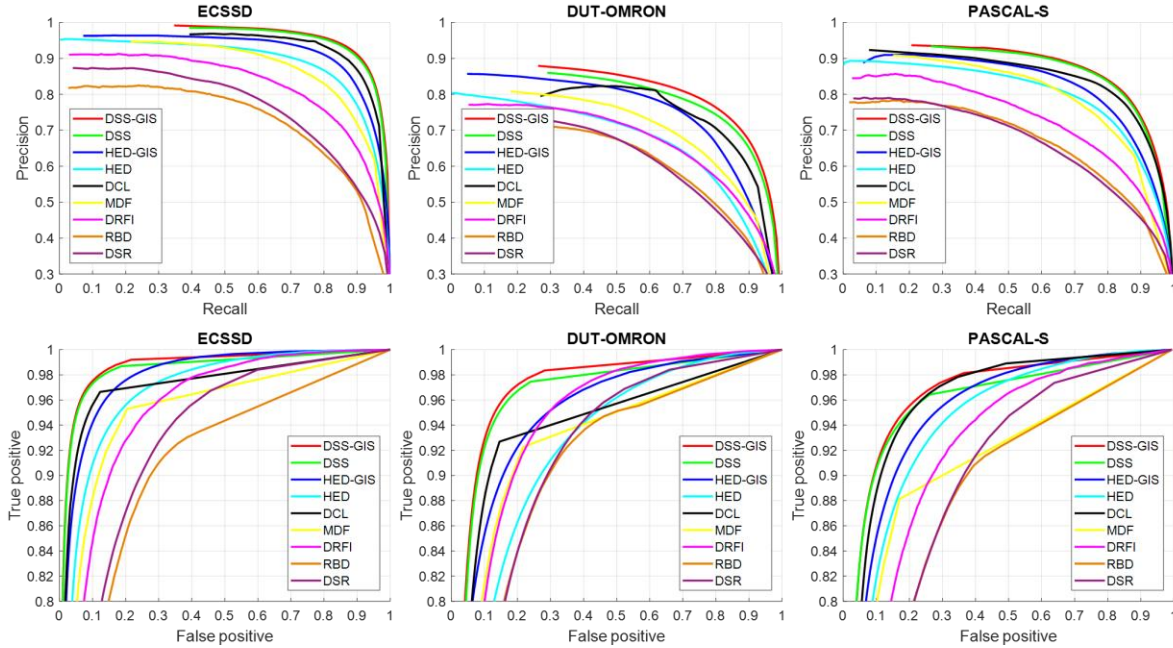


Figure 7-8 : Precision-Recall curves (top row) and true-positive/false-positive curves (bottom row) for the 3 test datasets

7.5.4 Resistance to noise and low-light

In the current section, we show that the proposed GIS layer allows the network to perform significantly better on the tests set when a high amount of noise is added, or when the brightness is significantly reduced.

To demonstrate it, we modify all images from the 3 testing sets by adding a 30% salt-and-pepper noise. We show in Table 7.5 that the GIS layer significantly improves the F_m and AUC metrics. Furthermore, Figure 7-5 allows to observe this major difference, with the GIS layer allowing the model to find objects that were almost invisible to the standard method.

On standard images, the GIS layer only improved the ECSSD F_m by 0.4%. However, on the noisy images, the improvement is 4.5%. On average for the DSS model, the GIS improves the F_m by 3.9% and the AUC by 3.0%. For the HED model, the average improvement is 8.1% on the F_m and 10.7% on the AUC .

Additionally, we can observe in Figure 7-10 how the F-measure of proposed DSS-GIS is more stable than the DSS method for different levels of noise. The stronger the noise, the greater the margin between DSS-GIS and DSS.

The margin of improvement of the proposed GIS layer in a noisy setting is highly significant. This shows again the better generalizability of the saliency models since GIS allows the network to focus on the general features instead of very local noise and texture.

We believe that the major improvement is due to textures being more affected by the noise than edges, which plays in favor of the models implementing the GIS layer.

Table 7.5 : Comparison of the DSS and HED approaches with and without the proposed GIS layer when a 30% salt-and-pepper noise is added to the test set. The best result of each column is highlighted in bold. The values are in percentages.

Dataset	ECSSD		DUT-OMRON		PASCAL-S	
Method	F_m	AUC	F_m	AUC	F_m	AUC
HED [66]	40.1	62.3	27.3	57.4	43.2	66.3
HED-GIS	50.6	73.0	39.0	71.6	45.3	73.4
DSS [26]	67.8	85.8	54.2	84.2	65.0	84.3
DSS-GIS	72.3	89.6	59.4	88.2	67.0	85.6

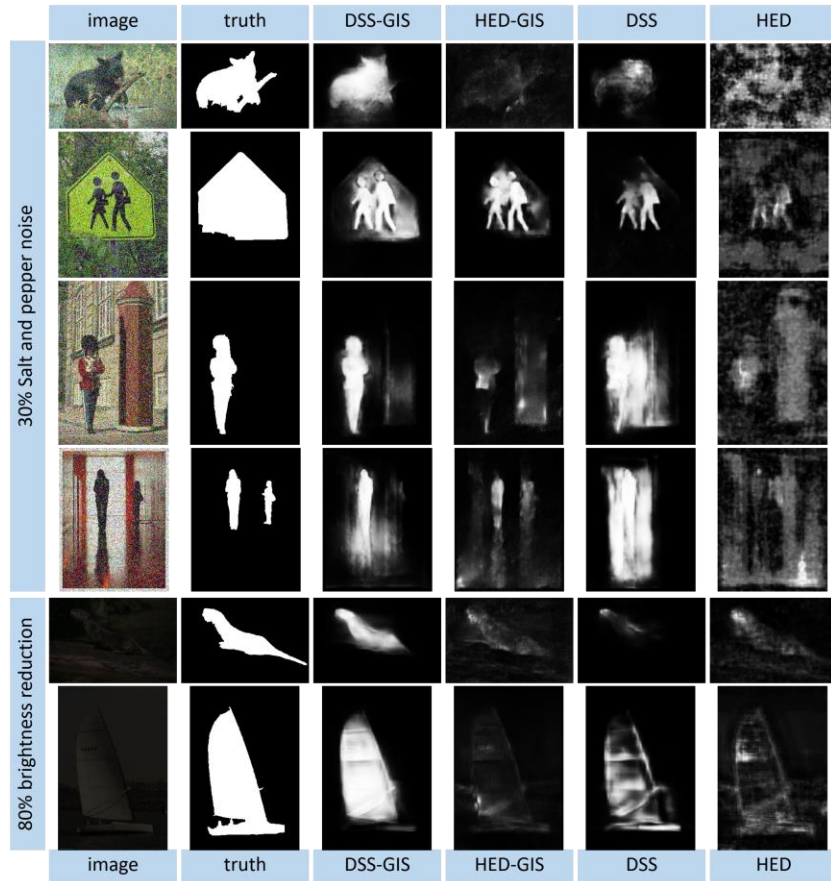


Figure 7-9 : Test results comparison between the DSS and HED methods with and without the proposed GIS layer, when the test set is modified with noise or reduced brightness.

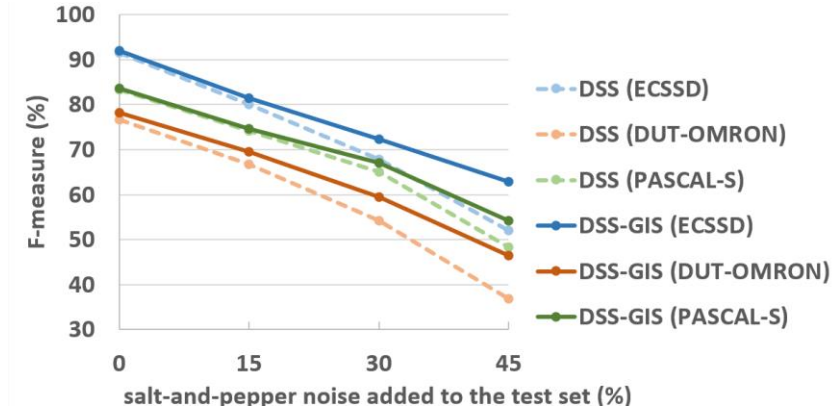


Figure 7-10 : Performance impact of adding noise to the testing set for the DSS and the proposed DSS-GIS methods.

Additionally to the added robustness to noise, the proposed GIS layer allows the network to be more robust to other environmental changes, such as reduction in brightness. This is demonstrated in Table 7.6 where DSS-GIS consistently outperforms DSS when the brightness is reduced by 80%, thus simulating an image taken at low light. Examples of such images are provided in Figure 7-9. This improvement is due to the ability of the GIS-based networks to operate in the gradient-domain and to extrapolate edge information.

Table 7.6 : Comparison of the DSS and HED approaches with and without the proposed GIS layer when a the brightness is reduced by 80% to simulate low-light pictures. The best result of each column is highlighted in bold. The values are in percentages.

Dataset	ECSSD		DUT-OMRON		PASCAL-S	
Method	F_m	AUC	F_m	AUC	F_m	AUC
HED [66]	68.2	87.4	53.2	83.6	60.9	84.2
HED-GIS	74.5	91.1	60.7	88.3	60.5	86.1
DSS [26]	83.0	93.3	69.8	91.1	74.4	88.2
DSS-GIS	84.5	95.8	72.6	94.4	76.9	91.4

7.5.5 Computation time

When using an image of the ECSSD dataset, the computation time for the DSS model is around 0.08s and the DSS-GIS is around 0.09s. Therefore, we see that the GIS layer has a low effect in

terms of computation cost. Moreover, it was shown in Figure 7-7 that it improves the training repeatability and in Table 7.3 and Figure 7-8 that it improves the F-measure on the testing results.

Other methods such as MDF [87] and DCL [76] require around 1s of computing due to the pre-segmentation, which is 10 times longer than the proposed DSS-GIS. Furthermore, the denseCRF layer takes around 0.4s to compute, which is 40 times longer than the added GIS layer. Hence, we suggest completely removing the denseCRF since it slows down the computation and leads to poorer performances than the GIS layer.

7.5.6 Future improvements

With the new GIS layer added at the end of the DSS network, the testing results are improved but by a moderate margin. One of the fundamental next steps is to take the same GIS layer, or other GFC-based layers, and to implement it deeper within the network, such as before the side layers or inside the VGGnet-16. Furthermore, the GIS layer should be tested for more applications such as classification, segmentation, and edge detection.

In fact, the GIS layers can be added to any other fully convolutional saliency architecture, not only the HED and DSS architecture as done in the current paper. Therefore, it adds good longevity to the GIS layer developed in this paper since newer architectures are also expected to benefit from the additional layer.

Finally, one of the most important contributions was showing that it is possible to add a Green's function convolution to a convolutional network to improve the results. This is surprising since CNN's usually have thousands of different and optimized convolutional kernels [9]. However, our work showed that a carefully engineered V_{mono} convolutional kernel can still contribute to improving the results. This is because V_{mono} adds a long-distance interaction between the pixel in the images, meaning that the receptive field is the whole image space. Also, since CNN are better at detecting edges than regions, integrating them into smooth and continuous regions naturally leads to better results.

For future work, we recommend using the same GIS network for segmentation purposes and for generative adversarial networks (GAN). In fact, we believe that the GIS layer would allow the GAN to generate image features in the gradient domain and the image domain at the same time. Since the GIS layer reduces noise sensitivity and gives the network an unlimited receptive field,

we strongly believe that it can help generate better images. Such gradient-domain image drawing is already adopted by numerous software to allow drawing smoother images [17,58,112,113].

7.6 Conclusion

Our objective was to show that saliency convolutional networks can be improved by using a Green's function convolution (GFC) based layer to extrapolate edges features into salient regions. To answer this objective, we developed the gradient integration and sum (GIS) layer. We showed that using a GIS layer, inside both HED and DSS neural networks, improves the stability and repeatability of the training and enhances the performance of the model on the test set, with only 10 ms of added computation time. The GIS layer is fast to compute and does not require any weight or learned parameter. Moreover, the GIS layer reduces the training time, the overfitting, and makes the model significantly more resistant to noise. The performance was generally better than other saliency enhancement methods such as denseCRF, 40 times faster to compute and directly integrated inside the network. Hence, our DSS-GIS network outperformed all the tested state-of-the-art algorithms on all tested metrics such as the F-measure, the *AUC* and the *MAE*. The increased performance was due to the ability of the network to extrapolate edges into regions, thus enhancing the saliency maps inside boundaries, reducing the sensitivity to noisy backgrounds and improving the behaviour in low-light settings. Further, an advantage of the proposed GIS is that it is model agnostic, meaning that it can be implemented in any other fully convolutional network for saliency. A limitation of the current method is that it can only be used in the latest layers of a fully convolutional network for saliency purposes. Future work should experiment with implementing the GIS layer or other GFC-based layers deeper inside the network to try to further improve the results.

CHAPTER 8 ADDITIONAL WORK

The current section will present additional work that is done in order to provide prototype and proof of concepts for the future works. These concepts include the creation of the GID2 and GID3 layers, which can be added into any CNN with limited computational cost. The early results show that the added layer allows the networks to learn faster and with higher accuracy than an identical network without the given layers.

8.1 Prototype and early results for the classification CNN

Due to the improvement of saliency results and the enabling of an unlimited receptive field, one of the hypotheses for future work is that the GFC will allow improving many different kinds of CNN for image analysis. To explore this idea, it is easier to start with the task of image classification since it is one of the most studied problems with the most straightforward CNN architectures [9].

Hence, we started by studying the effect of the GFC on a Google-net architecture applied on the MNIST [8,133] dataset. The MNIST dataset has 70,000 images of handwritten digits with a total of 10 classes (one per digit) [8,133]. It is one of the simplest classification datasets since digits are easier to classify than real images with an accuracy of 99.3% in 1998 [133]. The Google-net contains 6 consecutive inception layers and is usually used for more complex image classification [20]. For the simpler task of digits recognition, we only used 2 inception layers with 16 channels (also called neurons) per layer in the first inception module and 32 channels per layer in the second module. The networks are coded using Tensorflow® with an Adam optimizer [131], a learning rate of 10^{-4} and a batch size of 50 images.

To try to improve the results of the Google-net with the GFC, we developed the GID2 (Gradient Integration Derivative with 2-inputs) and the GID3 (Gradient Integration Derivative with 3-inputs). Then, we added a GID2 layer to each of the inception modules as shown by Figure 8-1. The GDI2 and GDI3 layers are explained in Figure 8-2. They both use a Conv-layer after their input without an activation function or bias, meaning that the layer is simply a linear combination of the previous layer. It is meant to allow negative values since the input is strictly positive due to the Relu activation.

The GID2 and GID3 are based on the idea that the features are a vector field similar to a gradient [9] or to Gabor functions which are mostly similar to Gaussian derivatives [9,41]. Hence, the

GID2-3 compute the derivatives dx and dy of the features to obtain a Laplacian. For the integration, the GID2 uses the GFC developed in section to solve the Laplacian, while the GID3 uses the GDM or GIS developed in section 6.2 to merge the gradient with the other features and then compute and solve the Laplacian. This is followed by the derivative step of the GID2-3 to compute a new vector field similar to the input one.

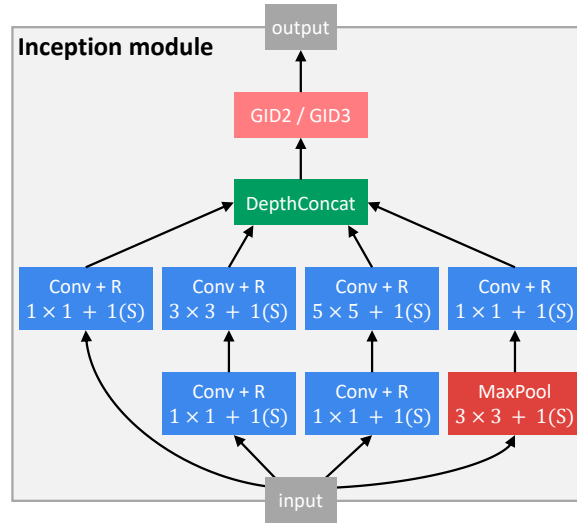


Figure 8-1 : Inception module of the Google-net [20] with an added GDI2 or GDI3 layer. The Conv + R indicate a convolutional layer with a Relu activation function, the $n \times n$ means that the kernel size is n and the $m(S)$ means that the stride is m .

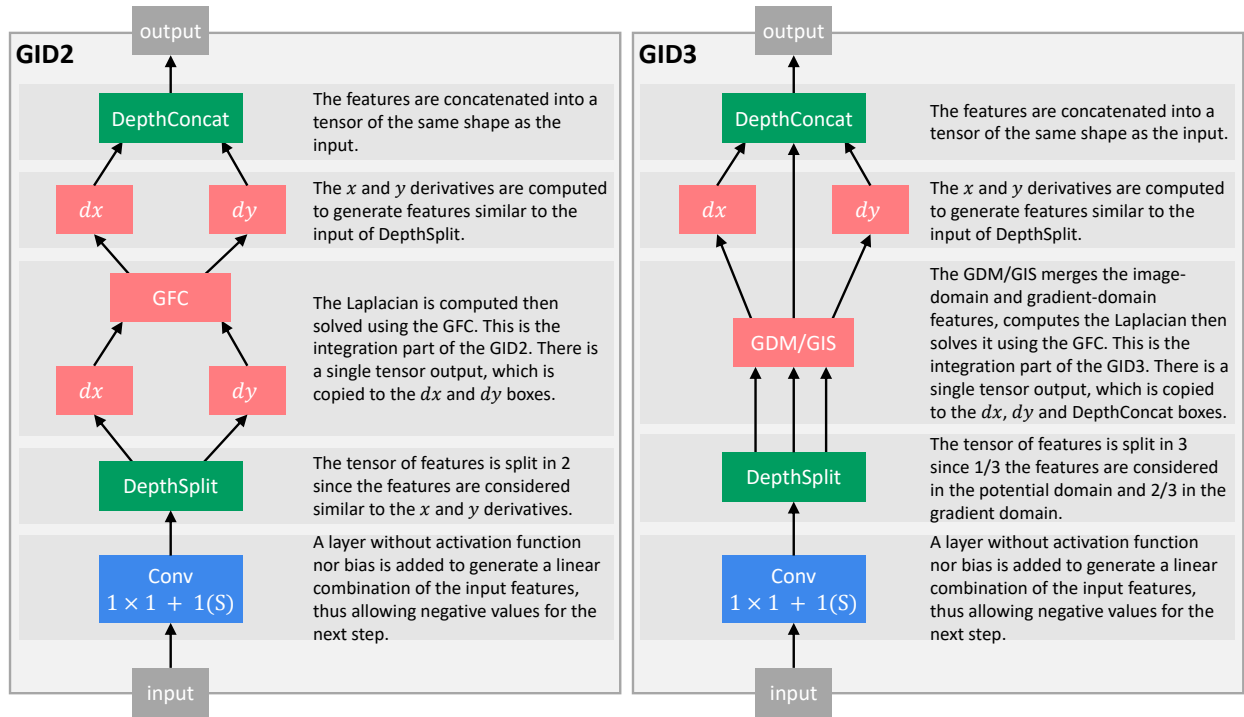


Figure 8-2 : Internal structure of the GDI2 which splits the data into 2 inputs for the GFC; and GDI3 which splits the data into 3 inputs for the GDM or GIS layers. The dx and dy represent numerical derivatives and the Conv-layer does not use an activation function or bias; the $n \times n$ means that the kernel size is n and the $m(S)$ means that the stride is m .

When testing the smaller Google-net composed of 2 inception modules on the MNIST dataset, we observe that the added GID2 significantly improves the results compared to the standard network, with all other parameters being exactly the same. The convergence to a 97% validation success rate is around 5.1 times faster with GID2. Furthermore, after 20,000 iterations, the model with the GID2 maxed to a smoothed validation accuracy of 98.80%, compared to 98.35% without the GID2, which is a 27% decrease in the error rate. These results are observed in Figure 8-3 where the orange line is strictly better (higher accuracy) than the blue line, especially for a low number of iterations. Furthermore, the orange line appears smoother than the blue line, meaning that it was easier for the gradient descent to converge to a minimum when using GID2. A downside of the added GID2 is that the training time is 2.0 times longer, but this still means that the convergence to 97% is 2.5 times faster than the standard network.

However, the GID3 did not improve the results, which can be due to the fact that handwritten digits do not have regional features such as saliency and the fact that regional features are only detected in deeper layers.

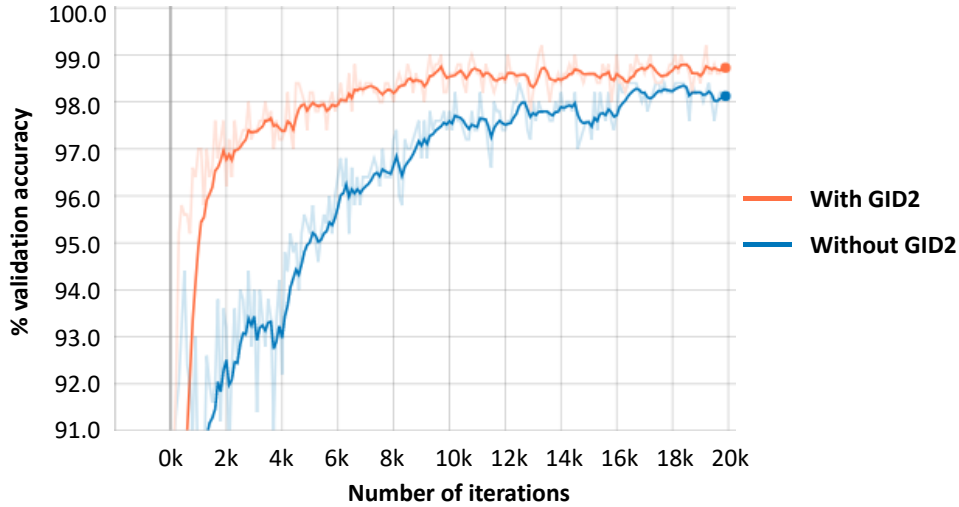


Figure 8-3 : Validation accuracy on the MNIST dataset for a smaller Google-net with 2 inception layers. The blue line is the standard network and the orange line is the same network with the added GID2 in each inception layer.

Before having a final verdict on the performance of the GID2, it is necessary to perform varied tests with deeper CNN and more complex images. However, we believe that GID2 will still be able to improve the results for many reasons. First, the integration with a Green's function followed by a derivative allows transforming the initial vector field of features into a conservative field, as demonstrated in Chapter 5. This conservative field has more continuous and smoother features with an unlimited receptive field between the given features.

We also believe that the conservative field features “make more physical sense”: for example, if we take a human picture and ask to identify the head, a non-conservative field could select an open shape, such as a $\frac{3}{4}$ of a circle. This does not make physical sense since the head feature should be a closed shape. For the conservative field, such open shape will be forced to close by a smooth gradient such as demonstrated in Chapter 4. Hence, the conservative feature field means that the features can be integrated into an existing potential solution.

Moreover, we know that detecting features in every possible orientation requires at least 2 kernels with non-colinear features, the same way that we need at least 2 vectors to generate the full 2D space. In that sense, the GID2 is very useful since it regularizes the features by encouraging half

of the features to be perpendicular to the other half to allow a proper integration with the GFC. All those reasons put together are believed to accelerate the optimization of the CNN by encouraging the gradient descent to follow a better optimization path.

In summary, we developed the GID2-3 which integrates the CNN features using a Green's function and derivates them back to give features represented by a conservative field. The GID2 was tested using a 2 inception layers Google-net on the MNIST handwritten digits dataset. The GID2 proved to reduce the final convergence error of 27%, requires 5.1 times fewer iterations to converge and has a smoother validation curve. However, more tests of the GID2 are required to verify if it helps improve the results with more complex networks and images.

8.2 Generative networks

In addition to using the GF for classification networks, we also believe the GF can be used for generative networks (GN) such as the popular generative adversarial networks (GAN) [134,135]. We expect the GF to have many advantages such as the regularization of the GN, the generation of a gradient, and the unlimited receptive field.

Regularizing the GN. Since the GF was shown to regularize a CNN by making it learn only features that are physically possible, we believe that it will help the GN focus on physically possible images. For example, a standard GN could generate a shape, but without properly closing its boundaries. However, using the GF, we believe that such an option will be avoided, thus improving the training of the GN.

Generating the gradient. Another aspect is that the GN could be used to generate the gradient of the image instead of generating the image. Then, the GF will be used to integrate the gradient into the desired image. This will mimic how humans generate images by drawing the contours of an object before filling it up. By using the same GIS layer as Chapter 7, the GN will be able to combine both gradient-domain and image-domain information for the generation.

Unlimited receptive field. Finally, the GF will enable unlimited receptive field between the pixels of the generated image, which is fundamental in ensuring that the pixels are generated based on global information.

CHAPTER 9 GENERAL DISCUSSION AND FUTURE WORK

This section will present a general discussion on how the proposed thesis answers the objectives given in section 1.1. Then, it will explain how the work distinguishes itself from the literature, the impact it has on the computation tools for machine vision, the improvement that it brings to deep neural networks and the future possible improvements. Finally, the section demonstrates the set of tools and deliverables that were done within the scope of the thesis.

9.1 Achieving the research objectives

The main objective of the current thesis is to **develop electromagnetic (EM) convolutions and Green's functions (GF) convolutions to be used in Computer Vision and convolutional neural networks (CNN).**

To understand how our thesis achieves the main objective, we will first explain how each of the sub-objectives is achieved.

Obj - 1. Develop a mathematical and intuitive understanding of the behavior of EM and GF convolutions in an image.

First, we developed the novel EM-based convolutional kernels in Chapter 3 where we studied the behavior of different dipole and monopole kernels in an image. In section 3.4.1, we studied how the EM potential and field behave when applied to different kind of 2D or 3D shapes. In section 3.4.2, we also explained how the dipoles can be used to fill-up a closed shape with a constant potential. Furthermore, the same technique was used to partially fill-up partial contours and make them interact with each other. Hence, Chapter 3 answered the objective by allowing to understand some geometrical properties of the EMPF and by developing an intuition on how to use them in an image.

This property of the dipoles was the main motivation for Chapter 4, where we demonstrated mathematically in section 4.3 that the dipole potential allows computing the space probability that any point is included inside a partial contour. This is very innovative in the CV field since it is the first work that allows such a transition from a 1D contour information to a 2D spatial information. In addition, some basic applications were shown in Figure 4-16 where one could approximate a simple image by using only the partial contours

corresponding to the highest gradients. Hence, Chapter 4 was important for the sub-objective by allowing to understand a new mathematical property of the EM potentials.

Finally, in Chapter 5 we realize that the EM potentials are a Green function (GF) which allows solving the Laplacian via a convolution. In section 5.2.2, we decided to study how the GF behaves when applied to a nonconservative field, meaning that the Laplacian does not have a solution. Hence, we prove mathematically that the GF convolution is the least-error solver for any nonconservative field. Therefore, Chapter 5 answers to the objective by mathematically demonstrating that the EM and GF based convolutions are optimal gradient and Laplacian solvers. Additionally, we show that the developed GFC method is orders of magnitude faster than competing approaches.

Obj - 2. Use the GF convolutions to reduce the computation time and numerical error of the EM and allow fast and efficient gradient-domain image editing (GDIE).

With the previous mathematical knowledge, we realize that the previous EM potential has a numerical error due to the discrete nature of an image. Hence, the section 5.2.3 develops a new ideal GF that reduces to almost zero the numerical error by using the Fourier domain to compute the convolutional kernel. This convolution is also faster to compute since the EM kernels need to be twice the size of the image, but the GF requires a kernel the same size as the image. In fact, our method was demonstrated in Figure 5-2 to be 16 times faster on a CPU and 3.1 times faster on a GPU than the fastest competing methods. In addition, the improved GF kernel keeps all the mathematical and geometrical properties developed in Chapter 3 and Chapter 4, but with better accuracy. For example, Figure 4-10 showed that the EM kernels did not produce perfect circular paths due to numerical error, but this problem is not present when using GF instead of EM. Therefore, the new GFC achieves the objective by reducing the computation time and the numerical error associated with the EM convolutions.

With these new mathematical properties, section 5.3.2 shows how to use the GFC for GDIE, with some applications such as Poisson blending, gradient thresholding, and edge contrast and blurring (known in later chapters as gradient domain merging). The performance was shown in Figure 5-4 to be slightly better than the competing Perez algorithm by being smoother on the edges of the blended image [57]. Hence, our work was again able to answer

the objective by developing a fast and efficient method of doing GDIE using the proposed GFC.

Obj - 3. Use the GF to improve the results of CNN for salient object detection and digit classification.

Using the previously developed GFC and the gradient domain merging (GDM) method, the proposed thesis was able to improve the image saliency maps using edges. At first, Chapter 6 presented the SEE method which allowed to merge the results of different saliency methods with the results of an edge detection CNN via a combination of image preprocessing and saliency postprocessing. The SEE method increased the saliency values inside the edges and reduced them outside the edges, thus improving F-measure on average 6.6 times more than competing methods on the ECSSD dataset and 3.4 times on the DUT-OMRON dataset as shown in Figure 6-10.

Although those results answered the objective, the computation time and the complexity of the algorithm were high. Also, the improvement over DSS, the best-tested method, were limited. Hence, we developed the GIS layer in Chapter 7 which is integrated directly inside the DSS and HED networks. As shown in Table 7.3, this increased the F-measure by a similar margin as the SEE method, but with a simpler algorithm consisting of an added GIS layer with a low computation cost (0.01s per image). Furthermore, the GIS layer showed in Figure 7-7 to improve the convergence time, improve the robustness to parameter initialization and reduce the overfitting of the model. Also, Figure 7-9 and Figure 7-10 showed that the GIS layer allowed to significantly improve the robustness to noise and to brightness reduction. All these improvements answer to the current objective by improving the performance of CNN for salient object detection.

Finally, the current thesis proposes future work on how the GFC can be used for different CNN. In fact, the later section 8.1 explains how we built such a prototype based on the architecture of the GoogLeNet [20] for the task of digits classification. The curves in Figure 8-3 show how our proposed approach reduced the convergence time and improved the accuracy of the network. Hence, the proposed prototype allows answering the current objective by improving the task of digit classification using GFC.

9.2 New computation tools for machine vision

The presented work is amongst the first ones to use electromagnetism (EM) and Green's functions (GF) for image analysis. As discussed in the literature review at section 2.1.2, the previous work involving EM-like fields were used for feature detection such as partial contour orientation [13] and gradient-based edge detection [14]. Furthermore, section 2.3.2 explained that, to our knowledge, GF are only used for solving the Laplacian for image filtering and gradient domain image editing. Hence, no other method uses GF for contour filling and machine learning purposes. In contrast, the presented work shows how EM and GF can be used for 2D or 3D shape analysis, for computing the space probability of inclusion, for faster gradient-domain image editing, for saliency/edges merging, and for improvement of the saliency accuracy and training convergence in deep neural networks.

Therefore, this work is original in its methodology and presented successful algorithms in different CV categories. In fact, it presented a new set of tools that can be used to analyze images in ways that were not previously possible. Chapter 3 showed how to build resolution invariant symmetrical and anti-symmetrical convolution kernels based on EM, thus allowing to analyze 2D and 3D shapes and the interaction between them. Using those anti-symmetrical kernels, Chapter 4 mathematically demonstrated the first method of computing the space of probability of inclusion within partial contours. This method allows to take a set of 1D thin edges or partial contours and to transform them into 2D regional information. Then, the work of Chapter 5 showed that these same kernels can be improved by computing the GF, which then allows to solve the gradient or Laplacian of an image with no error and, as shown in Figure 5-2, it is 16 times faster than the Tanaka method [119] on a CPU and 29,000 times faster when using a parallel solver on a GPU. The GFC was also demonstrated mathematically, then empirically at Figure 5-3, that it is the least-error solver for perturbation added on the gradient. Furthermore, the work of Chapter 5 showed that the EM kernels of the previous chapters should be replaced completely by the GF since they are more precise and faster to compute, although all the mathematical and geometrical properties of the EM-based kernels still hold true for the GF-based kernels.

In summary, the developed GF-based kernels have multiple mathematical and geometrical properties that can be used to analyze different shapes with resolution invariance, to extrapolate 1D partial contours into a 2D space and to solve Laplacian and gradients. Those properties are then

used to improve the results of deep neural networks for salient object discovery as discussed in the section below.

9.3 Improving deep networks

Using the rigorously developed mathematical and geometrical properties of GF, the current thesis then focused on improving the results of deep neural networks for saliency purposes. Moreover, the future work section 8.1 will show how GF can be used in different types of deep neural networks.

9.3.1 Improvement of the saliency network

With the work presented in Chapter 6, the main idea is that salient object detection methods can benefit from edge detection methods by enhancing the saliency maps inside boundaries and decreasing it outside the boundaries. Therefore, the RCF[25] edge detection method was used alongside different saliency methods such as DRFI[67], DCL[76] and DSS[26], and those methods were merged in the gradient domain using our proposed GF. The method proved successful and outperformed any other saliency improvement method in terms of F-measure. However, a limitation is that 2 different deep networks are required and need to be trained separately, which makes the parameter optimization more difficult and increases the computation time.

These limitations were the main motives to develop the DSS-GIS model proposed in Chapter 7. This model is heavily based on the DSS[26] saliency model, but instead of finding only region-like salient features, the model would also find gradient-like features and merge them together in the gradient domain using the proposed GF. Hence, a gradient integration and sum (GIS) layer is added at the end of each parallel branch of the DSS model. It is the first time that a custom convolutional kernel based on EM or GF is proposed to be used within any kind of neural network, which again highlights the originality of the current thesis. The proposed DSS-GIS model showed that it is more stable in training, less prone to overfitting, performs better on the test set and is more stable to noise. The reduction in overfitting is thought to be caused by the increase in the number of detected features, as well as the way those features are linked together. Furthermore, an improvement of performance is thought to be caused by a saliency map that has greater fidelity to the edges of the objects.

In summary, the GF convolutions (GFC) were demonstrated to improve the performance of a saliency-purposed deep neural network, first by post-processing the outputs of the networks but later by being directly integrated into the last layers of the networks.

9.3.2 Enabling unlimited receptive field

Another major reason that the GF improves feature recognition is that it enables an unlimited receptive field, thus allowing a long-distance interaction between different features. This interaction is possible via a convolutional neural network (CNN) by introducing a pooling to reduce the matrix size or by using very deep networks. For the pooling, a problem is the decreased image resolution, which reduces the details of the detected features. For the increased depth, if the features are separated by 100 pixels and the kernel sizes are 3x3, then the CNN will need at least $\frac{100}{3-1} = 50$ layers (if no pooling is used) [9]. This is a problem since a deeper network is harder to optimize [9]. For the proposed GF, the goal is not to replace those methods but to complement them by enabling an unlimited receptive field without increasing the depth or reducing the resolution. An intuition of this interaction was given in Figure 3-13 and Figure 4-12 where edges far away are still able to produce a region of interest between them.

A more concrete example of this unlimited receptive field is presented in Figure 9-1 where we observe how the GFC allows to fill up the salient region by simply finding the salient contours of the objects. The regular feature detection is unable to find the salient region since the receptive field is too small with a kernel size usually varying between 3x3 and 7x7 pixels [9,10,20,26]. The results are drawn using an image editing software meaning that they are not real results, but the image of Figure 3-13 and Figure 4-12 are useful in explaining how the unlimited receptive field is useful. Furthermore, this kind of feature filling was previously demonstrated with the bird image in Figure 7-5 and the statue image in Figure 6-7.

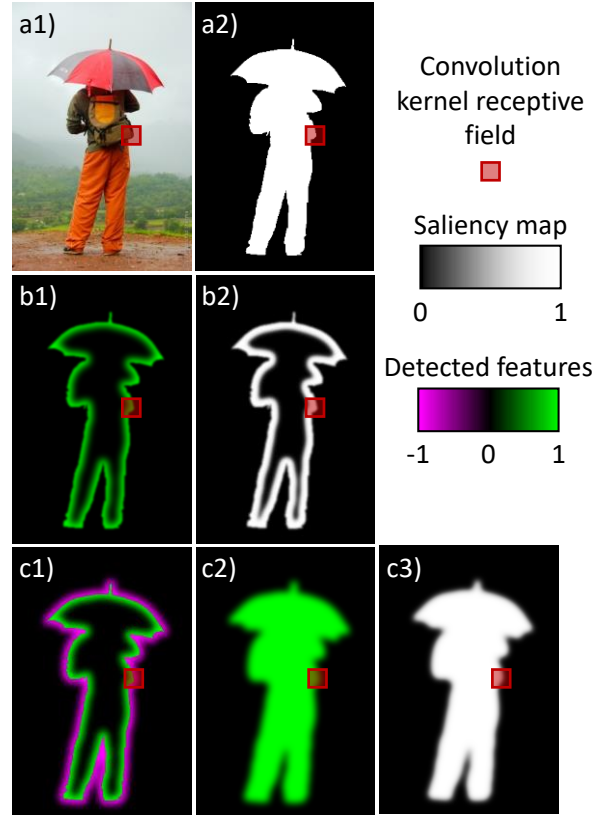


Figure 9-1 : Example of saliency maps produced with a single layer of convolution network which is able to perfectly detect the salient edges, but not the regions within the edges. The images are not real results, they are produced by an image editing software. (a1) Original image; (a2) True saliency value; (b1) Detected normalized saliency features with the small receptive field; (b2) Final saliency after thresholding; (c1) Detected normalized directional gradient of saliency features with the small receptive field; (c2) extended features using the GFC; (c3) Final saliency after GFC and thresholding.

9.4 Industrial applications and patents

As stated in Chapter 1, the current work is subject to 3 patents. The first one is published [21] and the second one planned for submission in March 2019. These 3 patents are created since the work of the current thesis is very innovative and has a lot of potential in industrial applications. Although industrial applications are not a necessity for academic research, they are still very important in engineering since they help the inventions reach the market and have an impact on the world.

The most developed aspect of the current thesis is the salient object detection described in Chapter 7, which can lead to many applications for smart cameras for auto-focus, selfie filters, and image

enhancement. In fact, some Huawei phones already use the DSS [26] model of salient object detection [132] on which our proposed DSS-GIS is based. Other applications could be for the automated removal of the background, such as proposed by the company 36Pix [136].

Another aspect described in Chapter 5 is the ability to do gradient-domain image editing (GDIE) (or Poisson image editing) for manual image/video editing purposes. Right now, the main method used is the Poisson solver proposed by Perez [57] which is available in standard computer vision libraries such as OpenCV. Some advanced gradient-domain libraries such as GradientShop [58] make extensive use of GDIE, which shows that many applications are available for manual image editing purposes. Our method proved to be 166 times faster than Perez on a CPU, and an additional 9 times faster on a GPU. This vast improvement will be very beneficial for real applications since high-resolution editing becomes instantaneous for an image and real-time for video applications.

Finally, the invention can be used to improve different kinds of neural networks in various fields. However, such application is only possible if the GFC is able to improve any kind of CNN, not just the saliency-purposed CNN, as discussed in section “9.5 Future work”.

9.5 Future work

With the major improvements brought by the added GFC in our DSS-GIS model, it is believable that the GFC can be implemented in different kind of deep network, not just the saliency-purposed. In fact, as discussed in section 9.3, the reasons that the GFC improves the network is that it enables the network to learn features and tasks in the image domain and the gradient domain at the same time, it enables unlimited receptive field at any layer, and it enables to transform any vector field of features into its nearest possible conservative field. Those properties can be beneficial in different kinds of CNN.

Therefore, a logical next step would be to test the GFC methods inside CNN with the purpose of edge detection, image segmentation or classification. An example is to implement the UFnet [11] which is a slightly modified version of the DSS [26] but with the added ability to do skeleton extraction and edge detection [11]. Furthermore, another important step would be to add the GFC at different depths within the network and verify if it helps to improve the network by forcing each step to merge region-like features with gradient-like features. For example, using the Google-net [20], one could try to implement the GFC inside each of the inception modules to verify if there is

an improvement as discussed in section 8.1. If the new approach proves to improve the results of standard classification networks, then the work developed in the current thesis could become an important part of the machine learning community. However, even if it fails, the GFC developed in this thesis will still be very useful for resolution invariant shape analysis, gradient-domain image editing, salient object detection and segmentation.

Furthermore, since we believe that the GFC layers allow the networks to better generalize to unseen data, we recommend to test them in a meta-learning or few-shot learning setting, where it is required to transfer knowledge from one task to others.

Other important work to be done is to package the whole code into libraries that can be easily implemented by industries for practical applications. Furthermore, the training set can be expanded to include a fraction of all the available saliency sets. For example, adding the HKU-IS dataset [87] proved to improve the results by up to 3% [11,132]. This work is fundamental to the success of the patents and to ensure their licensing, although it does not directly affect the academic results.

9.6 Limitations

The current thesis presented a novel way of using EM and GF for image analysis purposes. However, this section will present different mathematical limitations as well as practical limitations to the current work.

9.6.1 Mathematical limitations

The mathematical limitation concerns the type of neural network that can be used, the space on which the data is present and the computation complexity which limits the computation time.

9.6.1.1 Strictly convolutional networks

The first mathematical limitation is that the network must absolutely be convolutional to implement the GFC. This is because the methods that we develop assume that the features are similar to a gradient, which is translation invariant. Hence, a CNN is required since they are designed specifically to be translation invariant, contrarily to other methods such as standard NN and random forests [9]. Furthermore, since GFC is a convolution, it intuitively fits within a CNN since it has the same properties as the convolutional layers.

9.6.1.2 Euclidean space at least 2D

The second mathematical limitation is that the Laplacian and the Green's function are well defined in Euclidean space or a grid space such as a 1D vector, 2D image or 3D scan. However, although there are definitions of Fourier transforms and Laplacian on a graph [137], the computation changes if the graph is different [137]. This means that for graphs such as molecules which are different from each other, one cannot find a single Green's function.

Moreover, if the data is a 1D vector, the Green's function is useless since the integration is a straightforward cumulative sum of the vector. Hence, the developed method is not useful for music analysis or other 1D representation. This means that the method requires at least 2D to be useful.

Furthermore, the concept of a conservative feature field is not expected to work well in the case of sparse representation, where most values in a matrix are 0.

9.6.1.3 Computation complexity

Another mathematical limitation is the non-linear time complexity of the Fourier transform. For a fixed convolutional kernel size and an image of n pixels, the time complexity of a convolution is $O(n)$ but the time complexity of the Fourier transform is $O(n \log n)$ [23]. This means that for bigger images, the Fourier transform will slow down faster than the rest of the CNN. However, since an image has already a high number of pixels, the logarithm term is not too significant for the scaling. For example, if we scale a small image of 64×64 pixels to 128×128 pixels, a standard CNN requires 4 times more computation while a Fourier transform requires 4.67 times more computation, which is not too significant 17% increase.

Hence, the limitation is not too expensive computationally, but one should know about it when implementing a CNN with a GFC, GDM, GIS or GDI2-3 layer. In the case of the SEE algorithm implemented in Chapter 7, this is not a limitation since the GIS is applied on the layers with the least number of channels which keeps the computation time very low compared to the rest of the CNN. However, in the case of the GID2 layer applied inside a Google-net at section 8.1, the GID2 is applied on the layers with the highest number of channels. Therefore, time complexity becomes an important limitation and should be considered in the design of the network.

9.6.2 Practical limitations

The practical limitations consider the drawbacks of the method that is due to the way it is applied for image understanding: how it should not work well without gradient-like features, how it can produce undesired artifacts and how the performance of the method is not well understood.

9.6.2.1 Non-gradient-like features

The method was mainly developed to deal with gradient-like features such as the thin partial contours in Chapter 4, the gradient in Chapter 5, the salient edges in Chapter 6, the salient gradient in Chapter 7 and the gradient integration derivative in section 8.1. Therefore, the GFC is not expected to work when the features are not similar in nature to gradients. Although there is no demonstration that this statement is true, there is no indication that the GFC can be useful in other cases. Fortunately for the project, Goodfellow et al. [9] argue that the extracted visual features of both the visual cortex and inside the CNN are similar to the Gabor's function. Since most types of numerical Gabor's function are similar to the n th derivative of a Gaussian function, it counters well the current limitation.

9.6.2.2 Undesired artifacts

Although it was shown in section 5.2.2 that the Green's function is optimal at solving a perturbed gradient, Figure 5-8 shows that there can be some undesired artifacts when the gradient is too perturbed (or too far from being a conservative field). These kinds of artifacts were observed in the corner of the saliency maps during the first 2000 iterations of training the DSS-GIS model of Chapter 7. Fortunately, the artifacts disappear by themselves after more iterations. However, it is not clear if their effect is completely eliminated. Also, it is not clear if other networks such as the Google-net with GID2 developed in section 8.1 are affected internally by those artifacts since we do not directly observe the inner layers.

Adding a padding before the GFC was described in Chapter 5 as a mean to reduce the effect of those artifacts, but they do not eliminate them completely.

9.6.2.3 Unknown performance

Another practical limitation of the method is that the performance of the GF-based kernels in CNN is not well known. Should the GID2-3 layers developed in section 8.1 be used in deeper layers? In

shallower layers? In every inception layer? Should it skip some inception layers? Should it be used on all the inputs, or should there be some inputs not affected by the GID2-3? Does it work with complex images or only simpler images such as the MNIST digits? Does it work in another kind of networks or only in inception layers?

There are so many questions left unanswered as to when and how the current work performs well, and it is a major limitation for anyone that tries to implement GFC in another project. All those questions will require a few more years of practical research before they can be answered with confidence.

9.7 Thesis outcomes

For the current thesis work, we developed different tools and functionalities to close the contours and to use GFC within images or neural networks. To broaden the reach of our work, we decided to develop the tools in multiple languages/libraries, including Matlab, C++ (OpenCV) and Python (Pytorch and Tensorflow). This section summarizes briefly the tools and deliverables of the current thesis, followed by the scientific outcome.

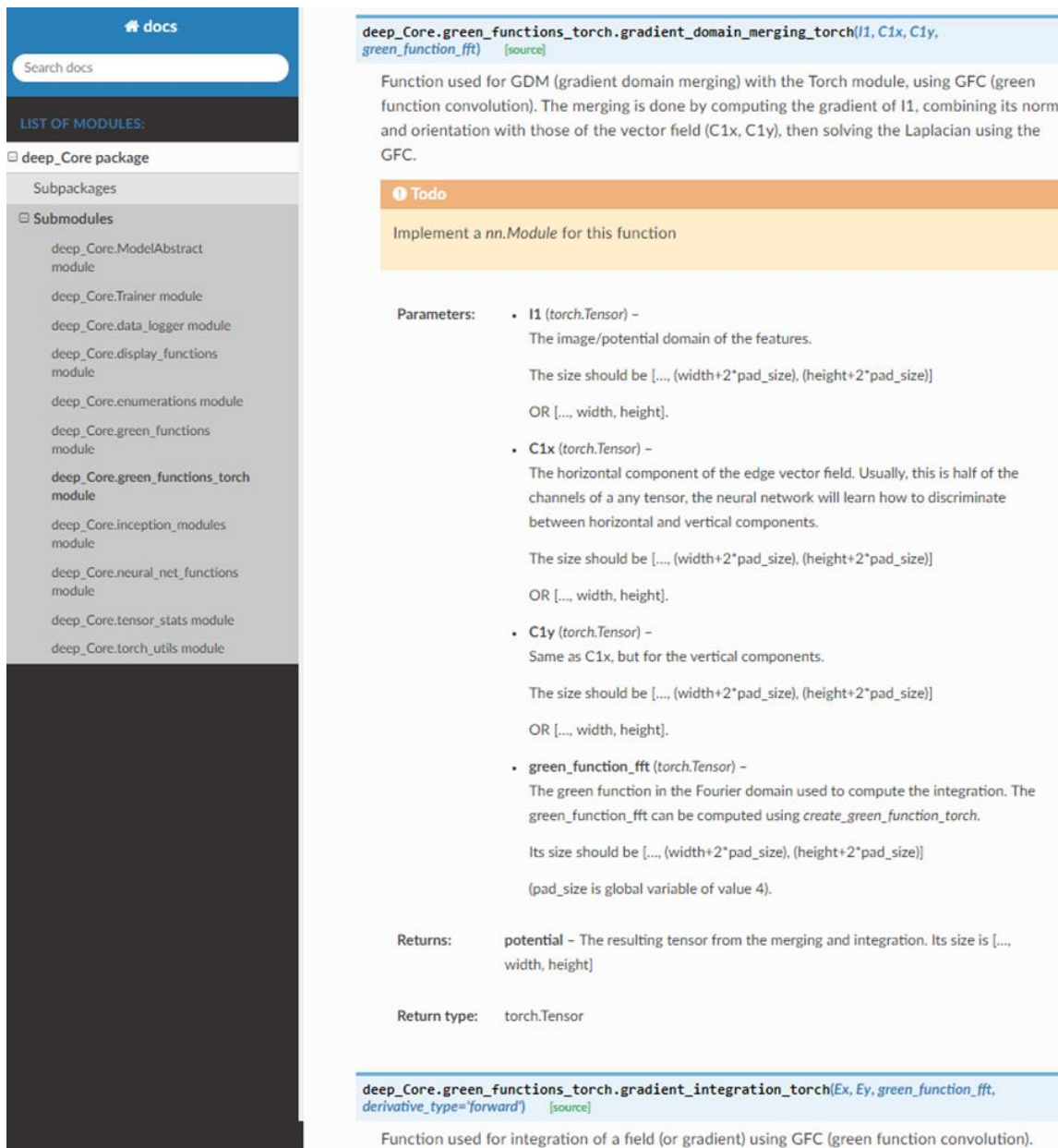
9.7.1 Tools

In Table 9.1, we can see the list of tools developed in each language and library. We observe that the older EM-based solvers were developed only in Matlab and OpenCV. However, the newer GF-based solvers are implemented in all languages. The GDM, GI, GIS and GID layers, useful for CNNs, are implemented in Matlab, as well as Tensorflow and PyTorch (2 of the most popular deep learning libraries).

Table 9.1 : Tools developed during the thesis in different languages and libraries

Tool	Matlab	OpenCV (C++)	Tensorflow (Python)	Pytorch (Python)
EM monopole (Laplacian solver)	✓	✓		
EM dipole (partial contour filling)	✓	✓		
GF monopole (Laplacian solver)	✓	✓	✓	✓
GF dipole (partial contour filling)	✓	✓		
GDM (gradient domain merging)	✓		✓	✓
GI (Gradient integration)	✓		✓	✓
GIS (Gradient integration and sum)		✓	✓	✓
GID (Gradient integration derivative)	✓	✓	✓	✓

We also observe in Figure 9-2 that time and effort were put into building a comprehensive documentation of the developed tools, classes and functions, thus enabling the project and tools to be used by other students, collaborators or researchers. The documentation was build using the Sphinx python package [138]. Additionally, each repository has its documentation, how-to section, and recommendations. For confidentiality reasons, the code is not yet publicly available, but could become publicly available in the future.



docs

Search docs

LIST OF MODULES:

▢ **deep_core package**

Subpackages

▢ **Submodules**

- deep_core.ModelAbstract module
- deep_core.Trainer module
- deep_core.data_logger module
- deep_core.display_functions module
- deep_core.enumerations module
- deep_core.green_functions module
- deep_core.green_functions_torch module**
- deep_core.inception_modules module
- deep_core.neural_net_functions module
- deep_core.tensor_stats module
- deep_core.torch_utils module

deep_core.green_functions_torch.gradient_domain_merging_torch(I1, C1x, C1y, green_function_fft) [\[source\]](#)

Function used for GDM (gradient domain merging) with the Torch module, using GFC (green function convolution). The merging is done by computing the gradient of I1, combining its norm and orientation with those of the vector field (C1x, C1y), then solving the Laplacian using the GFC.

Todo

Implement a `nn.Module` for this function

Parameters:

- I1** (`torch.Tensor`) – The image/potential domain of the features.
The size should be [..., (width+2*pad_size), (height+2*pad_size)]
OR [..., width, height].
- C1x** (`torch.Tensor`) – The horizontal component of the edge vector field. Usually, this is half of the channels of a any tensor, the neural network will learn how to discriminate between horizontal and vertical components.
The size should be [..., (width+2*pad_size), (height+2*pad_size)]
OR [..., width, height].
- C1y** (`torch.Tensor`) – Same as C1x, but for the vertical components.
The size should be [..., (width+2*pad_size), (height+2*pad_size)]
OR [..., width, height].
- green_function_fft** (`torch.Tensor`) – The green function in the Fourier domain used to compute the integration. The `green_function_fft` can be computed using `create_green_function_torch`.
Its size should be [..., (width+2*pad_size), (height+2*pad_size)]
(pad_size is global variable of value 4).

Returns: **potential** – The resulting tensor from the merging and integration. Its size is [..., width, height]

Return type: `torch.Tensor`

deep_core.green_functions_torch.gradient_integration_torch(Ex, Ey, green_function_fft, derivative_type='forward') [\[source\]](#)

Function used for integration of a field (or gradient) using GFC (green function convolution).

Figure 9-2 : Example of documentation used for our *green_function_torch* module.

9.7.2 Deliverables

An important part of the current thesis was to develop a tool that can work in real time on an embedded system for salient object detection. This tool can then be used on autonomous robots and cameras for better focus and better scene understanding.

For this purpose, there is an on-going project financed by NSERC-INNOV (National sciences and engineering research council of Canada, Idea to innovation grant). Using the Nvidia Jetson TX2,

an embedded GPU with CUDA support, we installed the Tensorflow library and were able to remotely compute the saliency maps in real-time using the DSS-GIS developed in Chapter 7. Future progress of this project is to develop real-time background removal and object detection for industrial applications such as assistive robotics and waste sorting.

9.7.3 Scientific outcomes

The current thesis has multiple scientific outcomes on mathematical and theoretical aspects, practical tools, concrete applications and scientific publications. In fact, the first 2 papers and first patent were about building a strong mathematical and theoretical background. The 3rd paper and 2nd patent were about building practical tools from the theory. Then, the last 2 papers, the last patent and the additional works were about building concrete applications of object discovery. Finally, the on-going Innov project is about delivering the academic technology to real-world applications.

Scientific publications. Firstly, there are a total of 8 scientific publications linked to the current paper, including 3 papers published on Arxiv, 2 papers in revision for scientific journals, 1 patent and 2 provisional patents. These publications demonstrate the originality and innovation brought by the current thesis.

Mathematical and theoretical aspects. In the first 2 papers and the first patent, we were able to innovate by finding new methods of extrapolating edges and contours to regions, bridging the gap between 2 fundamental computer vision approaches. We demonstrated mathematically that EM and GF-based kernels allow to determine the probability that any point is located inside a partial contour, which to our knowledge, was never done previously. We showed prototypes and simple examples to demonstrate how they can be used for object detection when incomplete information is given.

Practical tools. We showed in the 3rd paper and 2nd patent that we were able to solve Laplacian and non-conservative fields simply, with few lines of code with different computation libraries. In fact, we showed that 100 Laplacian can be solved in 1ms, which was orders of magnitude faster than competing approaches. For this work, we implemented the GF in Matlab, OpenCV and PyTorch, broadening reach of the developed tools.

Concrete applications. In the 4th and 5th paper, the 3rd patent and the additional work, we focused on bringing concrete and useful applications to the set of tools that we developed. With the DSS-

GIS method, major improvement of saliency detection were reported, especially for complex or noisy images, showing that the networks were able to better generalize outside the training data. Furthermore, the additional works showed that the GF-based tools can be used to improve an CNN with minimal architectural changes, simply by transforming the set of learned features into conservative features.

From academia to industry. As explained in section 9.7.2, important deliverables of the project is to make it ready for real-world and industrial applications. At the time of writing, there is a prototype of an embedded system that is able to detect salient objects in real-time, thus enabling future robotic applications.

Machine learning (A new CNN operation). The greatest contribution to the machine learning community is probably the creation of the GF-based operations, a new category of operations that can be implemented directly inside CNNs. These new operations enable completely novel properties, such as enabling the network to regularize the features by making them conservative (thus physically interpretable) or enabling the network to work simultaneously in the image and gradient-domain. An overview of this contribution is presented in Figure 9-3. It this to note that GF-based operations are convolutions, yet they are more similar to activation functions in how they behave. However, contrarily to activation functions, they perform non-trivial operations with the purpose of modifying or integrating a field of features.

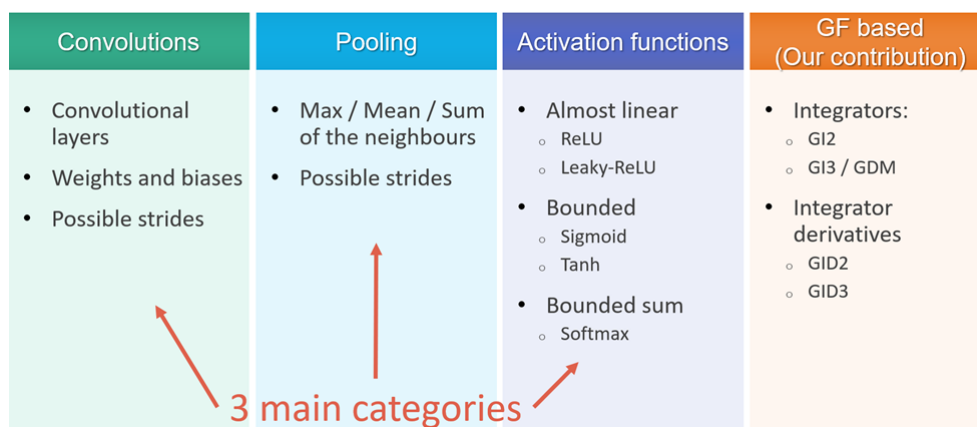


Figure 9-3 Our contribution, in terms of categories of possible operations inside CNN.

CHAPTER 10 CONCLUSION AND RECOMMENDATIONS

The main objective was to develop electromagnetic (EM) convolutions and Green's functions (GF) convolutions to be used in Computer Vision and convolutional neural networks (CNN). We believe to have successfully reached this objective, with a total of 2 theoretical computer vision papers, 1 practical paper in the field of image editing, and 2 practical papers in the field of salient object detection. In addition, one of the most important scientific outcomes of the thesis is the creation of a novel category of GF-based operations to be used in CNNs, which improves the training, the accuracy and the generalizability of the networks.

The first sub-objective was to develop a mathematical and intuitive understanding of the behavior of EM and GF convolutions in an image. This sub-objective was achieved in different aspects. First, Chapter 3 established how to create the EM kernels to generate resolution and rotation invariant convolutional kernel. It was later shown in Chapter 4 to be the only possible convolutional kernel that allows determining the probability of inclusion within open partial contours, thus allowing to extend edge information into spatial information. Chapter 5 followed by showing that GF-based kernels are an improved version of the EM-based kernels and by proving that GF kernels are the least-error gradient solver.

The second sub-objective was to use the GF convolutions to reduce the computation time and numerical error of the EM and allow fast and efficient gradient-domain image editing. The negligible error was demonstrated by testing on 1000 images and showing that GF has an average RMSE error of 0.011 on 256 grey levels, which is negligible. Furthermore, Figure 5-2 shows that the GF solves the Laplacian 16 times faster than competing methods on a CPU and 29,000 times faster when using multi-thread on a GPU. The implementation is also simpler since the algorithm is described entirely by the short pseudo-codes given in Algorithm 5-A and in Algorithm 5-B. In fact, the code uses the fast Fourier transform (FFT) without any loop or optimization.

The third sub-objective was to use the GF to improve the results of CNN for salient object detection and digit classification. This sub-objective was first solved with the SEE method proposed in Chapter 6, which uses the GF-based gradient domain merging (GDM) to merge saliency maps with edge detection. In Figure 6-10, it showed an improvement of the F-measure in average 6.6 times more than the nearest competing method on the ECSSD dataset. Then, Chapter 7 followed by developing the DSS-GIS method, which integrates the GIS layer directly inside the DSS. The added

GIS layer was demonstrated in Figure 7-7 to improve the convergence time, improve the robustness to parameter initialization and reduce the overfitting of the model. Finally, section 8.1 showed a promising prototype that enables using GF convolution inside image classification networks, thus reducing the convergence epochs by a factor 5.1, reducing the error by 27% and thus extending the reach of the current thesis. The GF-based layers, created in the present thesis, represent a new category of CNN operation which greatly differ from any other standard operation (weighted convolution, activation function and pooling). Hence, they are a major contribution of the thesis, especially since they improve the CNN training, accuracy and generalizability.

Consequently, the thesis contributes to the computer vision field by developing a new and performant computational tool based on GF that are fast to compute and proven to perform well in different applications such as gradient-domain image editing, salient object detection, and image classification. A major contribution is adding the category of GF-based operations to the neural network, thus enabling the network to work in the image and gradient-domain concurrently and enabling to transform any set of features into conservative and physically interpretable features. With these added GF-based layers, we observed significant reduction in the training time with a reduction of the hyperparameter sensitivity. Furthermore, the models were demonstrated to better generalize by improving the training accuracy for different architectures and by reducing the noise sensitivity.

However, there are some limitations to our work since GF can only be used in 2D+ convolutional networks, it has $O(n \log(n))$ time complexity and it was only shown to work on gradient-like features. Future work could focus on implementing the GF inside classification networks with more complex tasks and architectures. Also, a quantitative analysis of different GF-based layers should be done to optimize its usage. Moreover, we recommend doing a thorough analysis of how the GF affect the weights and the validation loss of different CNN architectures and for different datasets.

For future work, we recommend exploring the use of GF for generative networks, since the GF regularizes the networks for physically possible solutions and allows to generate images in the gradient domain. We also recommend making better use of the improved network generalization by exploring the use of GF for few-shot learning and meta-learning applications.

BIBLIOGRAPHY

- [1] Digital cameras: shipments by region 1999-2017 | Statistic. Statista n.d.
<https://www.statista.com/statistics/264338/shipments-of-digital-cameras-by-world-region-since-1999/> (accessed July 27, 2018).
- [2] Cell phone sales worldwide 2007-2017. Statista n.d.
<https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/> (accessed July 27, 2018).
- [3] Spending forecast - commercial robotics globally 2025 | Statistic. Statista n.d.
<https://www.statista.com/statistics/441964/forecast-for-commercial-robotics-spending-worldwide/> (accessed July 30, 2018).
- [4] Global computer vision AI market by industry application 2015-2019 | Statistic. Statista n.d.
<https://www.statista.com/statistics/641922/worldwide-artificial-intelligence-computer-vision-market-revenues/> (accessed July 30, 2018).
- [5] Gamache J-F, Vadean A, Noirot-Nérin É, Beaini D, Achiche S. Image-based truss recognition for density-based topology optimization approach. *Struct Multidiscip Optim* 2018;1–13. doi:10.1007/s00158-018-2028-x.
- [6] Christopher Bousquet-Jette, Sofiane Achiche, Dominique Beaini, Yann-Seing Law-Kam Cio, Cédric Leblond-Ménard, Maxime Raison. Fast scene analysis using vision and artificial intelligence for object prehension by an assistive robot. *Eng Appl Artif Intell* 2017;63:33–44. doi:10.1016/j.engappai.2017.04.015.
- [7] Leroux M, Raison M, Adadja T, Achiche S. Combination of eyetracking and computer vision for robotics control - IEEE Conference Publication n.d.
<https://ieeexplore.ieee.org/document/7219692> (accessed January 31, 2019).
- [8] MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges n.d.
<http://yann.lecun.com/exdb/mnist/> (accessed February 16, 2017).
- [9] Goodfellow I, Bengio Y, Courville A. *Deep Learning*. MIT Press; 2016.
- [10] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv14091556 Cs* 2014.

- [11] Hou Q, Liu J, Cheng M-M, Borji A, Torr PHS. Three Birds One Stone: A Unified Framework for Salient Object Segmentation, Edge Detection and Skeleton Extraction. ArXiv180309860 Cs 2018.
- [12] Hubel DH, Wiesel TN. Receptive fields and functional architecture of monkey striate cortex. J Physiol 1968;195:215–43.
- [13] Davis RS. Optical images by quadrupole convolution. US5027419 A, 1991.
- [14] Sun G, Liu Q, Liu Q, Ji C, Li X. A novel approach for edge detection based on the theory of universal gravity. Pattern Recognit 2007;40:2766–75. doi:10.1016/j.patcog.2007.01.006.
- [15] Beaini D, Achiche S, Cio Y-SL-K, Raison M. Novel Convolution Kernels for Computer Vision and Shape Analysis based on Electromagnetism. ArXiv180607996 Cs 2018.
- [16] Beaini D, Achiche S, Nonez F, Raison M. Computing the Spatial Probability of Inclusion inside Partial Contours for Computer Vision Applications. ArXiv180601339 Cs Math 2018.
- [17] Beaini D, Achiche S, Nonez F, Brochu Dufour O, Leblond-Ménard C, Asaadi M, et al. Fast and Optimal Laplacian and Gradient Solver for Gradient-Domain Image Editing using Green Function Convolution 2018.
- [18] Beaini D, Achiche S, Duperré A, Raison M. Saliency Enhancement using Gradient Domain Edges Merging n.d.
- [19] Beaini D, Achiche S, Duperré A, Raison M. Deep Gradient Domain Merging for Improved Saliency in Convolutional Networks n.d.
- [20] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going Deeper with Convolutions. Comput. Vis. Pattern Recognit. CVPR, 2015.
- [21] Dominique Beaini, Sofiane Achiche, Maxime Raison. Object analysis in images using electric potentials and electric fields. WO2018045472A1, n.d.
- [22] Gurney K. An Introduction to Neural Networks. CRC Press; 2003.
- [23] Szeliski R. Computer Vision: Algorithms and Applications. Springer Science & Business Media; 2010.
- [24] Russell SJ, Norvig P. Artificial Intelligence: A Modern Approach. Prentice Hall; 2010.

- [25] Cheng MM. Richer Convolutional Features for Edge Detection. 南开大学媒体计算实验室 2017. <https://mmcheng.net/rcfedge/> (accessed May 30, 2018).
- [26] Hou Q, Cheng M-M, Hu X-W, Borji A, Tu Z, Torr P. Deeply supervised salient object detection with short connections. *IEEE Trans Pattern Anal Mach Intell* 2018;1–1. doi:10.1109/TPAMI.2018.2815688.
- [27] Zhang P, Wang D, Lu H, Wang H, Ruan X. Amulet: Aggregating Multi-level Convolutional Features for Salient Object Detection. *ArXiv170802001 Cs* 2017.
- [28] Salman N. Image Segmentation Based on Watershed and Edge Detection Techniques. 2004.
- [29] Li B, Acton ST. Active Contour External Force Using Vector Field Convolution for Image Segmentation. *IEEE Trans Image Process* 2007;16:2096–106. doi:10.1109/TIP.2007.899601.
- [30] Maxwell JC. *A Treatise on Electricity and Magnetism*. Clarendon Press; 1881.
- [31] Feynman RP, B FRPSMLLR, Leighton RB, Sands M. *The Feynman Lectures on Physics, Desktop Edition Volume II: The New Millennium Edition*. Basic Books; 2013.
- [32] Rothwell EJ, Cloud MJ. *Electromagnetics, Second Edition*. CRC Press; 2008.
- [33] Arfken GB, Weber HJ. *Mathematical Methods for Physicists, 6th Edition*. 6th edition. Boston: Academic Press; 2005.
- [34] Forsyth D, Ponce J. *Computer Vision: A Modern Approach*. Pearson; 2012.
- [35] Laganière R. *OpenCV 2 Computer Vision Application Programming Cookbook*. Birmingham: Packt Publishing; 2011.
- [36] Corke P. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer Science & Business Media; 2011.
- [37] Kuhl FP, Giardina CR. Elliptic Fourier features of a closed contour. *Comput Graph Image Process* 1982;18:236–58. doi:10.1016/0146-664X(82)90034-X.
- [38] Calli B, Wisse M, Jonker P. Grasping of unknown objects via curvature maximization using active vision. 2011 *IEEEERSJ Int. Conf. Intell. Robots Syst. IROS*, 2011, p. 995–1001. doi:10.1109/IROS.2011.6094686.

- [39] Backes AR, Florindo JB, Bruno OM. Shape analysis using fractal dimension: a curvature based approach. *Chaos Interdiscip J Nonlinear Sci* 2012;22:043103. doi:10.1063/1.4757226.
- [40] Wang Z, Zhang D. Progressive switching median filter for the removal of impulse noise from highly corrupted images. *IEEE Trans Circuits Syst II Analog Digit Signal Process* 1999;46:78–80. doi:10.1109/82.749102.
- [41] Kumar A, Pang GKH. Defect detection in textured materials using Gabor filters. *IEEE Trans Ind Appl* 2002;38:425–40. doi:10.1109/28.993164.
- [42] Mak KL, Peng P, Yiu KFC. Fabric defect detection using morphological filters. *Image Vis Comput* 2009;27:1585–92. doi:10.1016/j.imavis.2009.03.007.
- [43] Li B, Acton ST. Vector Field Convolution for Image Segmentation using Snakes. 2006 Int. Conf. Image Process., 2006, p. 1637–40. doi:10.1109/ICIP.2006.312619.
- [44] Lenz I, Lee H, Saxena A. Deep learning for detecting robotic grasps. *Int J Robot Res* 2015;34:705–724.
- [45] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. *Adv. Neural Inf. Process. Syst.* 25, Curran Associates, Inc.; 2012, p. 1097–1105.
- [46] Romeny B ter H, Florack L, Koenderink J, Viergever M, editors. *Scale-Space Theory in Computer Vision*. Berlin Heidelberg: Springer-Verlag; 1997.
- [47] Ren X, Fowlkes CC, Malik J. Scale-invariant contour completion using conditional random fields. *Tenth IEEE Int. Conf. Comput. Vis. 2005 ICCV 2005*, vol. 2, 2005, p. 1214-1221 Vol. 2. doi:10.1109/ICCV.2005.213.
- [48] Achuthan A, Rajeswari M, Ramachandram D, Aziz ME, Shuaib IL. Wavelet energy-guided level set-based active contour: A segmentation method to segment highly similar regions. *Comput Biol Med* 2010;40:608–20. doi:10.1016/j.compbimed.2010.04.005.
- [49] Arbelaez P, Maire M, Fowlkes C, Malik J. Contour Detection and Hierarchical Image Segmentation. *IEEE Trans Pattern Anal Mach Intell* 2011;33:898–916. doi:10.1109/TPAMI.2010.161.

- [50] Qi Y, Song Y-Z, Xiang T, Zhang H, Hospedales T, Li Y, et al. Making Better Use of Edges via Perceptual Grouping, 2015, p. 1856–65.
- [51] Stahl JS, Wang S. Edge Grouping Combining Boundary and Region Information. *IEEE Trans Image Process* 2007;16:2590–606. doi:10.1109/TIP.2007.904463.
- [52] Ming Y, Li H, He X. Connected contours: A new contour completion model that respects the closure effect. 2012 IEEE Conf. Comput. Vis. Pattern Recognit. CVPR, 2012, p. 829–36. doi:10.1109/CVPR.2012.6247755.
- [53] Ming Y, Li H, He X. Contour Completion without Region Segmentation. *IEEE Trans Image Process* 2016;PP:1–1. doi:10.1109/TIP.2016.2564646.
- [54] Bui TD, Ahn C, Shin J. Unsupervised segmentation of noisy and inhomogeneous images using global region statistics with non-convex regularization. *Digit Signal Process* 2016;57:13–33. doi:10.1016/j.dsp.2016.06.002.
- [55] Zhao F, Fan J, Liu H. Optimal-selection-based suppressed fuzzy c-means clustering algorithm with self-tuning non local spatial information for image segmentation. *Expert Syst Appl* 2014;41:4083–93. doi:10.1016/j.eswa.2014.01.003.
- [56] Naik D, Shah P. A Review on Image Segmentation Clustering Algorithms. *IJCSIT* 2014;5:3289–93.
- [57] Pérez P, Gangnet M, Blake A. Poisson Image Editing. *ACM SIGGRAPH 2003 Pap.*, New York, NY, USA: ACM; 2003, p. 313–318. doi:10.1145/1201775.882269.
- [58] Bhat P, Zitnick CL, Cohen M, Curless B. GradientShop: A Gradient-domain Optimization Framework for Image and Video Filtering. *ACM Trans Graph* 2010;29:10:1–10:14. doi:10.1145/1731047.1731048.
- [59] Jeschke S, Cline D, Wonka P. A GPU Laplacian Solver for Diffusion Curves and Poisson Image Editing. *ACM SIGGRAPH Asia 2009 Pap.*, New York, NY, USA: ACM; 2009, p. 116:1–116:8. doi:10.1145/1661412.1618462.
- [60] Orzan A, Bousseau A, Barla P, Winnemöller H, Thollot J, Salesin D. Diffusion Curves: A Vector Representation for Smooth-shaded Images. *Commun ACM* 2013;56:101–108. doi:10.1145/2483852.2483873.

- [61] Tanaka M, Kamio R, Okutomi M. Seamless Image Cloning by a Closed Form Solution of a Modified Poisson Problem. SIGGRAPH Asia 2012 Posters, New York, NY, USA: ACM; 2012, p. 15:1–15:1. doi:10.1145/2407156.2407173.
- [62] Borji A, Cheng M-M, Hou Q, Jiang H, Li J. Salient Object Detection: A Survey. ArXiv14115878 Cs Q-Bio 2017.
- [63] Canny J. A Computational Approach to Edge Detection. IEEE Trans Pattern Anal Mach Intell 1986;PAMI-8:679–98. doi:10.1109/TPAMI.1986.4767851.
- [64] Dollár P, Zitnick CL. Fast Edge Detection Using Structured Forests. IEEE Trans Pattern Anal Mach Intell 2015;37:1558–70. doi:10.1109/TPAMI.2014.2377715.
- [65] Dollar P, Zitnick CL. Structured Forests for Fast Edge Detection, 2013, p. 1841–8.
- [66] Xie S, Tu Z. Holistically-Nested Edge Detection 2015.
- [67] Jiang H, Wang J, Yuan Z, Wu Y, Zheng N, Li S. Salient Object Detection: A Discriminative Regional Feature Integration Approach, IEEE; 2013, p. 2083–90. doi:10.1109/CVPR.2013.271.
- [68] Shi J, Yan Q, Xu L, Jia J. Hierarchical Image Saliency Detection on Extended CSSD. IEEE Trans Pattern Anal Mach Intell 2016;38:717–29. doi:10.1109/TPAMI.2015.2465960.
- [69] Borji A, Cheng M-M, Jiang H, Li J. Salient Object Detection: A Benchmark. IEEE Trans Image Process 2015;24:5706–22. doi:10.1109/TIP.2015.2487833.
- [70] Fu H, Cao X, Tu Z. Cluster-Based Co-Saliency Detection. IEEE Trans Image Process 2013;22:3766–78. doi:10.1109/TIP.2013.2260166.
- [71] Li X, Lu H, Zhang L, Ruan X, Yang MH. Saliency Detection via Dense and Sparse Reconstruction. 2013 IEEE Int. Conf. Comput. Vis., 2013, p. 2976–83. doi:10.1109/ICCV.2013.370.
- [72] Lu Y, Zhang W, Lu H, Xue X. Salient Object Detection using concavity context. 2011 Int. Conf. Comput. Vis., 2011, p. 233–40. doi:10.1109/ICCV.2011.6126247.
- [73] Perazzi F, Krähenbühl P, Pritch Y, Hornung A. Saliency filters: Contrast based filtering for salient region detection. 2012 IEEE Conf. Comput. Vis. Pattern Recognit., 2012, p. 733–40. doi:10.1109/CVPR.2012.6247743.

- [74] Zhu W, Liang S, Wei Y, Sun J. Saliency Optimization from Robust Background Detection, 2014, p. 2814–21.
- [75] Liu N, Han J. DHSNet: Deep Hierarchical Saliency Network for Salient Object Detection, 2016, p. 678–86.
- [76] Li G, Yu Y. Deep Contrast Learning for Salient Object Detection, 2016, p. 478–87.
- [77] Patel D, Raman S. Saliency Map Improvement Using Edge-Aware Filtering. *Comput. Vis. Pattern Recognit. Image Process. Graph.*, Springer, Singapore; 2017, p. 209–19. doi:10.1007/978-981-13-0020-2_19.
- [78] Tang L, Li H, Wu Q, Ngan KN. Boundary-Guided Optimization Framework for Saliency Refinement. *IEEE Signal Process Lett* 2018;25:491–5. doi:10.1109/LSP.2018.2801821.
- [79] Krähenbühl P, Koltun V. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *ArXiv12105644 Cs* 2012.
- [80] Annum R, Riaz MM, Ghafoor A. Saliency detection using contrast enhancement and texture smoothing operations. *Signal Image Video Process* 2018;12:505–11. doi:10.1007/s11760-017-1186-4.
- [81] DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs - *IEEE Journals & Magazine* n.d. <https://ieeexplore.ieee.org/abstract/document/7913730> (accessed October 5, 2018).
- [82] MNIST Demos on Yann LeCun’s website n.d. <http://yann.lecun.com/exdb/lenet/> (accessed October 18, 2018).
- [83] What is the VGG neural network? - *Quora* n.d. <https://www.quora.com/What-is-the-VGG-neural-network> (accessed October 9, 2018).
- [84] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. *ArXiv151203385 Cs* 2015.
- [85] CNN Architectures — LeNet, AlexNet, VGG, GoogLeNet and ResNet n.d. <https://medium.com/@RaghavPrabhu/cnn-architectures-lenet-alexnet-vgg-googlenet-and-resnet-7c81c017b848> (accessed August 14, 2019).

- [86] Tsang SH. Review: Inception-v4 — Evolved From GoogLeNet, Merged with ResNet Idea (Image Classification). Data Sci 2018. <https://towardsdatascience.com/review-inception-v4-evolved-from-googlenet-merged-with-resnet-idea-image-classification-5e8c339d18bc> (accessed February 24, 2019).
- [87] Li G, Yu Y. Visual Saliency Based on Multiscale Deep Features. ArXiv150308663 Cs 2015.
- [88] Paulinas M, Ušinskas A. A Survey of Genetic Algorithms Applications for Image Enhancement and Segmentation. Inf Technol Control 2015;36. doi:10.5755/j01.itc.36.3.11886.
- [89] Zheng H, Kong LX, Nahavandi S. Automatic inspection of metallic surface defects using genetic algorithms. J Mater Process Technol 2002;125–126:427–33. doi:10.1016/S0924-0136(02)00294-7.
- [90] Costa L da FD, Cesar RM Jr. Shape Analysis and Classification: Theory and Practice. 1st ed. Boca Raton, FL, USA: CRC Press, Inc.; 2000.
- [91] Loncaric S. A survey of shape analysis techniques. Pattern Recognit 1998;31:983–1001. doi:10.1016/S0031-2023(97)00122-2.
- [92] Backes AR, Bruno OM. A Graph-Based Approach for Shape Skeleton Analysis. Image Anal. Process. – ICIAP 2009, Springer, Berlin, Heidelberg; 2009, p. 731–8. doi:10.1007/978-3-642-04146-4_78.
- [93] Pratt I. Shape Representation Using Fourier Coefficients of the Sinusoidal Transform. J Math Imaging Vis 1999;10:221–35. doi:10.1023/A:1008396607943.
- [94] Smach F, Lemaître C, Gauthier J-P, Miteran J, Atri M. Generalized Fourier Descriptors with Applications to Objects Recognition in SVM Context. J Math Imaging Vis 2008;30:43–71. doi:10.1007/s10851-007-0036-3.
- [95] Fornberg B. Generation of finite difference formulas on arbitrarily spaced grids. Math Comput 1988;51:699–706. doi:10.1090/S0025-5718-1988-0935077-0.
- [96] Malik J, Perona P. A computational model of texture segmentation. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 1989 Proc. CVPR 89, 1989, p. 326–32. doi:10.1109/CVPR.1989.37868.

- [97] Batool N, Chellappa R. Detection and Inpainting of Facial Wrinkles Using Texture Orientation Fields and Markov Random Field Modeling. *IEEE Trans Image Process* 2014;23:3773–88. doi:10.1109/TIP.2014.2332401.
- [98] Dharmagunawardhana C, Mahmoodi S, Bennett M, Niranjana M. Gaussian Markov random field based improved texture descriptor for image segmentation. *Image Vis Comput* 2014;32:884–95. doi:10.1016/j.imavis.2014.07.002.
- [99] Gong M, Liang Y, Shi J, Ma W, Ma J. Fuzzy C-Means Clustering With Local Information and Kernel Metric for Image Segmentation. *IEEE Trans Image Process* 2013;22:573–84. doi:10.1109/TIP.2012.2219547.
- [100] Sabuncu MR, Yeo BTT, Leemput KV, Fischl B, Golland P. A Generative Model for Image Segmentation Based on Label Fusion. *IEEE Trans Med Imaging* 2010;29:1714–29. doi:10.1109/TMI.2010.2050897.
- [101] Kumar A, Nguyen L, DeGraef M, Sundararaghavan V. A Markov random field approach for microstructure synthesis. *Model Simul Mater Sci Eng* 2016;24:035015. doi:10.1088/0965-0393/24/3/035015.
- [102] Li C, Wand M. Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis, 2016, p. 2479–86.
- [103] Ulyanov D, Lebedev V, Vedaldi A, Lempitsky V. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn. - Vol. 48*, New York, NY, USA: JMLR.org; 2016, p. 1349–1357.
- [104] Wang H, Oliensis J. Generalizing edge detection to contour detection for image segmentation. *Comput Vis Image Underst* 2010;114:731–44. doi:10.1016/j.cviu.2010.02.001.
- [105] Zhang H, Fritts JE, Goldman SA. Image segmentation evaluation: A survey of unsupervised methods. *Comput Vis Image Underst* 2008;110:260–80. doi:10.1016/j.cviu.2007.08.003.
- [106] Chakraborty S, Mitra P. A dense subgraph based algorithm for compact salient image region detection. *Comput Vis Image Underst* 2016;145:1–14. doi:10.1016/j.cviu.2015.12.005.

- [107] Zhang Q, Liu Y, Zhu S, Han J. Salient object detection based on super-pixel clustering and unified low-rank representation. *Comput Vis Image Underst* 2017;161:51–64. doi:10.1016/j.cviu.2017.04.015.
- [108] Sagiv M, Reps T, Wilhelm R. Parametric Shape Analysis via 3-valued Logic. *ACM Trans Program Lang Syst* 2002;24:217–298. doi:10.1145/514188.514190.
- [109] Karpathy A, Miller S, Fei-Fei L. Object discovery in 3D scenes via shape analysis. 2013 IEEE Int. Conf. Robot. Autom. ICRA, 2013, p. 2088–95. doi:10.1109/ICRA.2013.6630857.
- [110] Werner T, Martín-García G, Frintrop S. Saliency-Guided Object Candidates Based on Gestalt Principles. In: Nalpantidis L, Krüger V, Eklundh J-O, Gasteratos A, editors. *Comput. Vis. Syst.*, Springer International Publishing; 2015, p. 34–44. doi:10.1007/978-3-319-20904-3_4.
- [111] Gonzalez Vivo P, Lowe J. *The Book of Shaders*. n.d.
- [112] McCann J, Pollard NS. Real-time Gradient-domain Painting. *ACM SIGGRAPH 2008 Pap.*, New York, NY, USA: ACM; 2008, p. 93:1–93:7. doi:10.1145/1399504.1360692.
- [113] Sun T, Thamjaroenporn P, Zheng C. Fast Multipole Representation of Diffusion Curves and Points. *ACM Trans Graph* 2014;33:53:1–53:12. doi:10.1145/2601097.2601187.
- [114] Sun X, Xie G, Dong Y, Lin S, Xu W, Wang W, et al. Diffusion Curve Textures for Resolution Independent Texture Mapping. *ACM Trans Graph* 2012;31:74:1–74:9. doi:10.1145/2185520.2185570.
- [115] Ilbery P, Kendall L, Concolato C, McCosker M. Biharmonic Diffusion Curve Images from Boundary Elements. *ACM Trans Graph* 2013;32:219:1–219:12. doi:10.1145/2508363.2508426.
- [116] Weisstein EW. Projection Theorem n.d. <http://mathworld.wolfram.com/ProjectionTheorem.html> (accessed June 8, 2018).
- [117] PyTorch n.d. <https://www.pytorch.org> (accessed June 14, 2019).
- [118] Afifi M. Poisson image editing - File Exchange - MATLAB Central 2017. <https://www.mathworks.com/matlabcentral/fileexchange/62287> (accessed July 13, 2018).

- [119] Tanaka M. Fast seamless image cloning by modified Poisson equation - File Exchange - MATLAB Central 2016. <https://www.mathworks.com/matlabcentral/fileexchange/39438> (accessed August 13, 2018).
- [120] TechPowerUp. TechPowerUp n.d. <https://www.techpowerup.com/gpu-specs/> (accessed June 24, 2019).
- [121] Sorna A, Cheng X, D'Azevedo E, Won K, Tomov S. Optimizing the Fast Fourier Transform Using Mixed Precision on Tensor Core Hardware. 2018 IEEE 25th Int. Conf. High Perform. Comput. Workshop HiPCW, 2018, p. 3–7. doi:10.1109/HiPCW.2018.8634417.
- [122] Cheng MM, Mitra NJ, Huang X, Torr PHS, Hu SM. Global Contrast Based Salient Region Detection. IEEE Trans Pattern Anal Mach Intell 2015;37:569–82. doi:10.1109/TPAMI.2014.2345401.
- [123] Dollar P. Structured Edge Detection Toolbox. Contribute to pdollar/edges development by creating an account on GitHub. 2018.
- [124] Yan Q, Xu L, Shi J, Jia J. Hierarchical Saliency Detection. 2013 IEEE Conf. Comput. Vis. Pattern Recognit., 2013, p. 1155–62. doi:10.1109/CVPR.2013.153.
- [125] Li Y, Hou X, Koch C, Rehg JM, Yuille AL. The Secrets of Salient Object Segmentation. ArXiv14062807 Cs 2014.
- [126] Yang C, Zhang L, Lu H, Ruan X, Yang M-H. Saliency Detection via Graph-Based Manifold Ranking, IEEE; 2013, p. 3166–73. doi:10.1109/CVPR.2013.407.
- [127] Bylinskii Z, Judd T, Oliva A, Torralba A, Durand F. What do different evaluation metrics tell us about saliency models? ArXiv160403605 Cs 2016.
- [128] Zhao R, Ouyang W, Li H, Wang X. Saliency detection by multi-context deep learning. 2015 IEEE Conf. Comput. Vis. Pattern Recognit. CVPR, 2015, p. 1265–74. doi:10.1109/CVPR.2015.7298731.
- [129] Hu P, Shuai B, Liu J, Wang G. Deep Level Sets for Salient Object Detection. 2017 IEEE Conf. Comput. Vis. Pattern Recognit. CVPR, Honolulu, HI: IEEE; 2017, p. 540–9. doi:10.1109/CVPR.2017.65.

- [130] Kuen J, Wang Z, Wang G. Recurrent Attentional Networks for Saliency Detection. 2016 IEEE Conf. Comput. Vis. Pattern Recognit. CVPR, Las Vegas, NV, USA: IEEE; 2016, p. 3668–77. doi:10.1109/CVPR.2016.399.
- [131] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. ArXiv14126980 Cs 2014.
- [132] Hou Q. Deeply Supervised Salient Object Detection with Short Connections - Applications. 南开大学媒体计算实验室 2018. <https://mmcheng.net/dss/> (accessed December 12, 2018).
- [133] Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE 1998;86:2278–324. doi:10.1109/5.726791.
- [134] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative Adversarial Nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Adv. Neural Inf. Process. Syst. 27, Curran Associates, Inc.; 2014, p. 2672–2680.
- [135] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ArXiv151106434 Cs 2015.
- [136] 36Pix – Perfect Keying n.d. <https://www.36pix.com/> (accessed December 12, 2018).
- [137] Bresson X. Convolutional Neural Networks on Graphs n.d.:87.
- [138] Sphinx - Python Documentation Generator n.d. <http://www.sphinx-doc.org/en/master/> (accessed August 17, 2019).
- [139] Hahnloser RHR, Sarpeshkar R, Mahowald MA, Douglas RJ, Seung HS. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. Nature 2000;405:947–51. doi:10.1038/35016072.

APPENDICES

APPENDIX A BENCHMARKING PARAMETERS

To properly compare and evaluate different saliency algorithms, the literature defined different standard datasets and metrics. For the datasets, MSRA10K [122], an extension of the previous MSRA-B [67] dataset, is the most widely used for training purposes. It is used for training since it has the largest number of images (10,000) and it is one of the easiest which makes it less appropriate for benchmarking [26].

For evaluation purposes, 3 other datasets are used: ECSSD with 1000 images [68,124], PASCAL-S [125] with 850 more complex images and DUT-OMRON with the most complex 5168 images [126].

For the benchmarking, the standard parameters that are evaluated are the precision P , the recall or true positives R and the false positives 1R [69,127]. Those parameters are evaluated for 256 levels of thresholds on the saliency map S , which allows to plot the precision-recall PR curve. At each threshold level, a binary mask M is generated and compared to the binary ground-truth G . From the PR curve, one can evaluate the average \overline{PR} , the F-measure F_m and the maximum precision P_{\max} . All those parameters are defined in equations (112)-(116), where $\beta = 0.3$ is a constant that allows to add more weight to the precision, 1 is the logical NOT operator, \cap is the logical AND operator and \sum is the sum over every pixel [69,127].

$$P = \frac{\sum M \cap G}{\sum M} \quad (112)$$

$$R = \frac{\sum M \cap G}{\sum G}, \quad ^1R = \frac{\sum M \cap ^1G}{\sum ^1G} \quad (113)$$

$$P_{\max} = \max(P) \quad (114)$$

$$F_m = \max\left(\frac{(1 + \beta^2)(P R)}{\beta^2 P + R}\right) \quad (115)$$

$$\overline{PR} = \int P dR \quad (116)$$

Other important information is the area under the curve (AUC) of the true-false-positive curve, and the mean absolute error (MAE) given respectively in equations (117) and (118), where S is the saliency map normalized to $[0, 1]$ and N the total number of pixels.

$$AUC = \int R \, d^1R \quad (117)$$

$$MAE = \frac{1}{N} \sum |S - G| \quad (118)$$

From all those parameters, the most used in the literature are the precision-recall PR curve, the F-measure F_m and the mean absolute error MAE . However, a problem with PR curves is that it is can be difficult to compare different curves together when the results are close and rank the different algorithms. Also, the MAE is easy to alter by changing the contrast of the saliency map. Hence, F_m is the most important measure and it is usually the one used for ranking the algorithms.

APPENDIX B STANDARD NEURAL NETWORKS

Neural networks are strongly inspired by how the brain works by connecting each neuron to a set of previous neurons and subsequent neurons. Each of the neurons performs simple operations, composed of the weighted sum and an activation function, based on their respective inputs from the previous neuron layer. Then, each neuron outputs the result to the next neuron layer [9]. This is explained visually in Figure 10-1 where the neurons are the yellow circles and are connected to the previous and subsequent layers.

The first operation is done by the neuron n_i is to compute W_i , the sum of the weighted input given in equation (119), where each input $x_{i,j}$ is multiplied by a weight $w_{i,j}$ [9]. The second operation adds a bias and uses an activation function α to compute the output y of the given neuron, as given by equation (120). The parameters $w_{i,j}$ and b_i are optimized by the neural network during the learning phase. The activation function is pre-determined before the learning and are biologically inspired to allow the NN to learn non-linear features [139]. Examples of the RELU and sigmoid activation functions are given in equations (121) and (122) [9].

$$W_i = \sum_j w_{i,j} x_{i,j} \quad (119)$$

$$y = \alpha(W_i + b_i) \quad (120)$$

$$\alpha_{RELU}(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (121)$$

$$\alpha_{sigmoid} = \frac{1}{1 + e^{-x}} \quad (122)$$

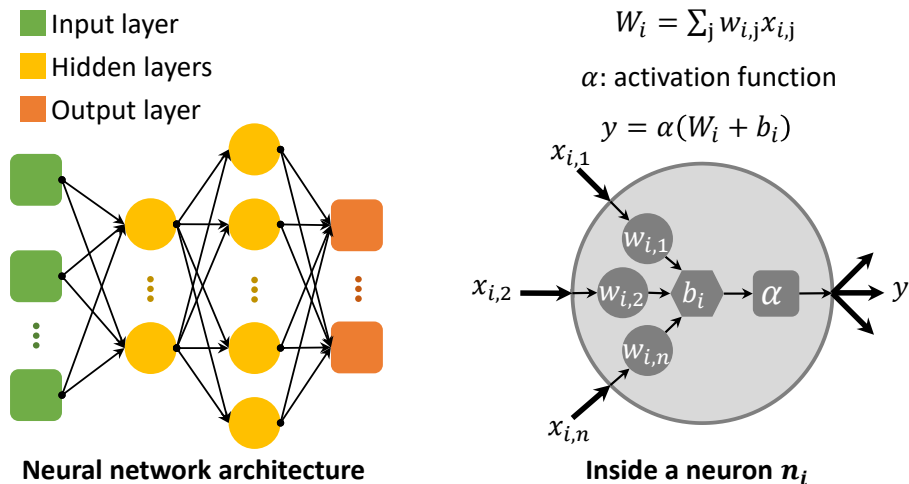


Figure 10-1 : Neural network architecture with the neurons being the yellow circles, with a view inside the neurons

In the NN of Figure 10-1, we can see that there is a variable number of neurons in each layer. However, the numbers of inputs and outputs are dependant on the application. For the example of classification, the output is the class of the object. The inputs can be raw data or extracted features. The depth of an NN is measured as the number of hidden layers, and deep NN is those with a depth higher than 1.

In the case of image understanding applications, the NN did not perform well on complex images since the raw data of each pixel would create a neural network too big and difficult to optimize. Hence, a human was often required to manually extract the features before feeding them to the NN.

APPENDIX C GENERAL ELECTROMAGNETISM

C.1 Supplementary Nomenclature

The following nomenclature is useful for the appendices, in addition to the nomenclature presented at the beginning of the paper.

$q_{e,m}$	Charge [C] _e , [A m] _m
$d_{e,m}$	Dipole charge separation [m]
r	Distance from an electric charge [m]
ϵ_0	Permittivity of free space [F m ⁻¹]
μ_0	Permeability of free space [N A ⁻²]
$\rho_{e,m}$	Density of charge [C m ⁻³] _e , [A m ⁻²] _m
J	Electric current [C s ⁻¹]
$\nabla \cdot$	Divergence operator
$\nabla \times$	Curl operator

C.2 Monopoles and Dipoles

C.2.1 Electric monopoles

Static electric monopoles are the most primitive elements that generate an electrical field, and they can be positive or negative. The positive charges generate an outgoing electric field and a positive potential, while the negative charges generate an ingoing electric field and a negative potential. This is shown in Figure 10-2, where the color scale is the normalized value of the electric potential \mathcal{V}_e and the arrows represent the electric field \mathcal{E}_e . In our 3D universe, the values of the potentials and fields of static charges are given by equation (14) [30–32]. However, the current paper will not limit itself to the 3D equations of electromagnetism, and more general equations will be developed.

$$\begin{aligned}\mathcal{V}_e &= \frac{q_e}{4\pi\epsilon_0\|r\|} \\ \mathcal{E}_e &= \frac{q_e}{4\pi\epsilon_0\|r\|^2} \hat{r}\end{aligned}\tag{123}$$

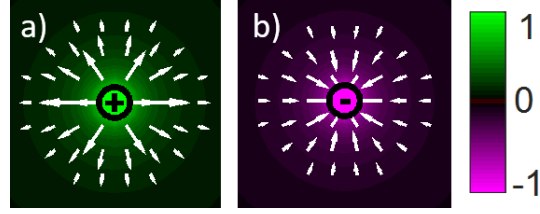


Figure 10-2 : Static electric potential and field of a: (a) positive monopole. (b) negative monopole

The color-bar used for the potential is shown in Figure 3-1 but will be omitted in many other images for concision. It is normalized so that the value “1” is associated with the maximum potential and “−1” is associated with the maximum negative potential.

When we deal with more than one particle, then the total potential and field is the sum of all the individual potentials and fields, as given by equation (16) [30–32]. It should be noted that the total potential is a simple scalar sum, while the total field is a vector sum.

$$\mathcal{V}_e^{tot} = \sum_i^n \mathcal{V}_e^i, \quad \boldsymbol{\mathcal{E}}_e^{tot} = \sum_i^n \boldsymbol{\mathcal{E}}_e^i \quad (124)$$

C.2.2 Electric dipoles

An electric dipole is created by placing a positive charge near a negative charge. This generates an electric potential that is positive on one side (positive pole), negative on the other side (negative pole) and null in the middle. The charge separation \boldsymbol{d}_e a vector corresponding to the displacement from the positive charge to the negative charge, and is mathematically defined at equation (16) [32].

$$\boldsymbol{d}_e = \boldsymbol{r}_{e+} - \boldsymbol{r}_{e-} \quad (125)$$

The electric field will then have a preferential direction along the vector \boldsymbol{d}_e by moving away from the positive charge, but it will loop back on the sides to reach the negative charge. Many examples of electric dipoles are presented at Figure 10-3, with the simplest form being composed of 2 opposite charges. On this figure, we notice that stacking multiple dipoles in a chain will not result in a stronger dipole, because all the positive and negative charges in the middle will cancel each other. Therefore, stacking the dipoles in series will only place the poles further away from each other. However, stacking the dipoles in parallel will result in a stronger potential and field on each

side of the dipole. It is also possible to see that the field will be almost perpendicular to the line of parallel dipoles, but it is an outgoing field on one side and an ingoing field on the other.

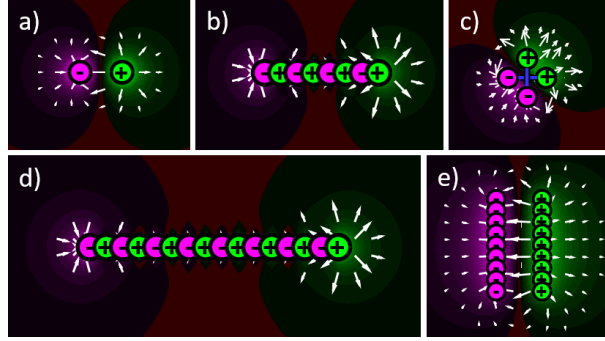


Figure 10-3 : Electric Potential and field for static monopoles placed as (a) A simple dipole. (b) A small chain of simple dipoles. (c) A horizontal and a vertical dipole, equivalent as 2 dipoles at 45° . (d) A long chain of simple dipoles in series. (e) A long chain of simple dipoles in parallel.

To calculate the total electric potential and field of any kind of dipole, it is possible to use the equation (14), without forgetting to change the sign of q_e accordingly. This sign change leads to a potential that diminishes a lot faster for dipoles at Figure 10-3 when compared to the monopoles at Figure 10-2. In a 3D world, with $\theta = 0$ alongside vector \mathbf{d}_e , the dipole potential will vary according to $\mathcal{V}_{dip} \propto \cos(\theta) / \|\mathbf{r}\|^2$, compared to the monopole potential which varies in proportion to $\mathcal{V}_e \propto 1/\|\mathbf{r}\|$ [31,32].

Another important aspect of dipoles is that when \mathbf{d}_e is small, the potential of a diagonal dipole is calculated by the linear combination of a horizontal and a vertical dipole. The potential of a dipole at angle θ (\mathcal{V}_{dip}^θ) is approximated by equation (17) [31,32]. This is easy to prove by using the previous statement that $\mathcal{V}_{dip} \propto \cos(\theta)$.

$$\mathcal{V}_{dip}^\theta \approx \mathcal{V}_{dip}^x \cos(\theta) + \mathcal{V}_{dip}^y \sin(\theta) \quad (126)$$

C.2.3 Magnetic charges and dipoles

Electricity and magnetism are 2 concepts with an almost perfect symmetry between them and will lead to similar mathematical equations. First, a magnetic dipole is what is commonly called a “magnet”, and is composed of a north pole (N) and a south pole (S). When compared to the electrical dipole, the north pole is mathematically identical to the positive pole and the south pole

is identical to the negative pole. Therefore, the potentials and fields of magnetic dipoles are identical to those of Figure 3-2, and the equations are the same as those defined by equations (14), except for the constants.

One can also mathematically define a magnetic monopole the same way as the electric monopole was defined. Although magnetic monopoles are not found in nature, nothing prevents us from using their mathematical concepts for computer vision.

C.3 Mathematical Laws of EM

C.3.1 Maxwell's Equations

The development of traditional electromagnetism was completed by J.C. Maxwell and allows to explain all the EM phenomenon using 4 mathematical equations, known as Maxwell's equations (MEq), which can be written with integrals or differential form. The first MEq is the Gauss law presented at equation (18) [30,32]. It means that the electric field that leaves a certain volume is directly proportional to the total charge inside it. The second MEq is Gauss law of magnetism and is presented at equation (19) [30,32]. It is identical to equation (18), except that the charge density is zero due to the inexistence of magnetic charges.

$$\nabla \cdot \mathcal{E}_e = \frac{\rho_e}{\epsilon_0} \quad (127)$$

$$\nabla \cdot \mathcal{E}_m = 0 \quad (128)$$

The following MEq are known as Faraday's law (20) and Ampere's law (21). They allow understanding the behavior of EM when there are time variations of the fields [30,32].

$$\nabla \times \mathcal{E}_e = -\frac{\partial \mathcal{E}_m}{\partial t} \quad (129)$$

$$\nabla \times \mathcal{E}_m = \mu_0 \left(J + \epsilon_0 \frac{\partial \mathcal{E}_e}{\partial t} \right) \quad (130)$$

Another important concept in EM is the electrostatic potential $\mathcal{V}_{e,m}$, which is a scalar defined as the line integral of the field $\mathcal{E}_{e,m}$, given by equation (22) [32].

$$\mathcal{V}_{e,m} = - \int_C \boldsymbol{\varepsilon}_{e,m} \cdot d\mathbf{l} \quad (131)$$

C.3.2 Adaptation of Maxwell's equations for Computer Vision

The important equations of EM were developed in the previous section, but their current form is not adapted for computer vision. First, the presence of the constants ε_0 and μ_0 are not useful for the current application. It is also possible to ignore the fact that magnetic charges cannot exist and regroup equations (18) and (19) to generate equation (20). Furthermore, the time variation and the current from equations (20) and (21) are ignored to generate equation (24). Thus, the 4 MEq are simplified into 2 new equations for static electromagnetism given by (23) and (24). These equations are the same for Electricity and Magnetism. Also, there is no interaction between a static electric field and a static magnetic field [30,31]. For these reasons, the current paper will often use the term “electric” when using monopoles and “magnetic” or “magnetize” when using dipoles, because it is more intuitive.

Equation (23) means that the total virtual field going out of a surface is directly proportional to the number of virtual charges contained inside. For dipoles, the total virtual field is null because the sum of charges is always zero. Equation (24) means that there is no curl to the field. By using equation (24) with equation (22), it is possible to demonstrate equation (25) [32], which states that the field is given by the gradient of the potential.

$$\nabla \cdot \mathbf{E}_{e,m} = \rho_{e,m} \quad (132)$$

$$\nabla \times \mathbf{E}_{e,m} = 0 \quad (133)$$

$$\mathbf{E}_{e,m} = -\nabla V_{e,m} \quad (134)$$

With these equations demonstrated, the next step is to determine the potential and the fields that are generated by charged particles. By using equation (23) and by knowing that, at a certain radius, the field around a single particle is uniformly spread, it is possible to show that equation (3) holds. The variable “ n ” denotes the dimension of the universe where the potentials and fields are used. This means that for a 3D universe we have $\mathbf{E}_{e,m} \propto 1/|r|^2$, for a 2D universe, we have $\mathbf{E}_{e,m} \propto 1/|r|^1$, while for a 1D universe, $\mathbf{E}_{e,m}$ is constant. This is in concordance with the real laws of

electromagnetism, if we acknowledge that a 2D universe is when the infinite wire approximation is used, and that a 1D universe is when an infinite plate approximation is used. However, the current paper will not limit equation (3) to a finite universe by choosing non-integer values for n . Therefore, we have that $E_{e,m} \propto 1/|r|^{n-1}$, for a universe of n spatial dimensions.

C.4 Geometrical Interpretation of Maxwell's Equations

C.4.1 Closed shapes with particles on the contour

If we have a closed circle composed of charged particles on the contour, then equation (23) allow to see that the field will be almost null inside the circle, but really high as soon as we are outside the circle. The potential is strong and constant in the middle of the circle, but diminishes rapidly outside the circle [31,32], as observed at Figure 10-4. This is because the potential is scalar, therefore the contribution of each particle will be summed. However, the field requires a vector sum, which means that they will cancel each other in the middle as the vectors will be of opposite direction, but they will add themselves outside of the circle. This holds true for any kind of closed shape, although the perfect symmetry of a circle makes the cancelation of the field more effective.

By using a circle of dipoles instead of monopoles, with the radii being R_- and $R_+ > R_-$, then the field will be null and the potential almost constant everywhere, except for a position R that respects the inequality $R_- < R < R_+$. This is due to the gauss law (23) and is observed at Figure 10-4. It also holds true for any kind of closed shape with dipoles on all of its contour.

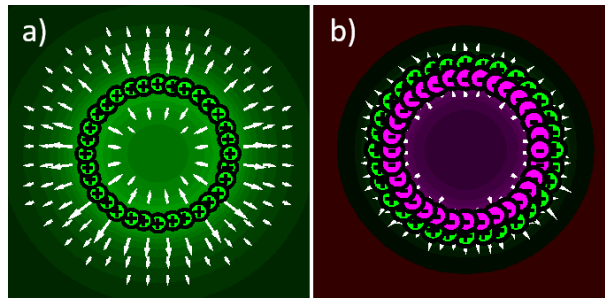


Figure 10-4 : Potential and field for circles, with $n = 3$, for (a) positive monopoles. (b) parallel dipoles. The green part is the positive charges/potential, and the purple part is the negative charges/potential.

C.4.2 Corners with particles on the contour

If we have a corner composed of charged particles as presented in Figure 10-5, then it is possible to analyze the potential and field and to see that it somewhat resembles the closed circle. The concave part of the corner will have a low field due to the vector sum of opposite vectors, but it will be the point with the highest potential, just like the inside of the circle. The convex part of the corner will have a slightly higher field because the vectors are less destructive, but the potential will be a lot lower because it is further away from the other charges, similarly to the exterior of the circle. Finally, the flat parts of the corner will have the highest field, but average potential.

In Figure 10-5, it is possible to see that the field on the corners will have a diagonal direction due to the contribution of both sides of the corner. For the dipole corner, the behavior is really similar than the monopole corner, except that the concave part has an ongoing field with a negative potential, while the convex part has an outgoing field with positive potential. It should be noted that the field will tend to be at an angle of 45° when it is far at the top-left or bottom-right of the corner.

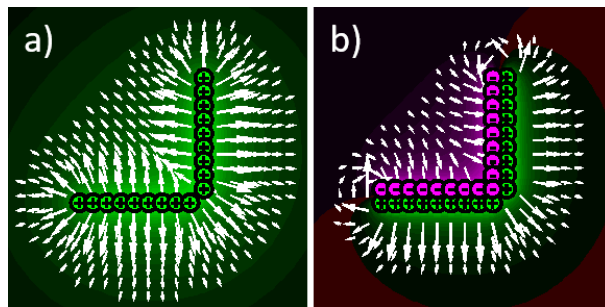


Figure 10-5 : Potential and field for corners of (a) positive monopoles. (b) parallel dipoles.

C.5 Partial contour Manipulations

This section presents algorithms on how to manipulate the partial contours or contours of an image and how to grow/shorten specific regions from the contour. The pseudocodes make use of some Matlab® functions for computer vision, but they all have their equivalent in other image processing libraries such as OpenCV®.

C.5.1 Grow and unite contour regions

A contour region is defined as a group of pixels that are part of the contour. For example, the high potential region will be everywhere on the contour with a high value of V_e . However, due to discretization, the regions might have discontinuities. Also, it could be required to simply grow the desired regions by a specific number of pixels at each side of the region, but by keeping it on the contour.

In the current paper, the regions are always grown by a percentage of the biggest dimension of an image (%BL). This allows to always be consistent no matter the resolution or the scale of the image. The regions are expended using a loop of image dilation. The dilations will expand the region one pixel in all directions, and then be multiplied by the contours to remove the undesired growth. This process is detailed in Algorithm 10-A.

Algorithm 10-A. Pseudocode for the growth of a region on the contour

```
// growthPercentage: Desired %BL for the growth
// regionOnContour: Matrix with value 1 on parts of the contour to grow
// contour: Matrix with value 1 on the contour
Function GROW_REGION(growthPercentage, regionOnCoutour, contour)
// Find the number of pixels to grow
numberPixelToGrow = round(growthPercentage * max(size(image)));
// Compute the geodesic distance between each point on the contour and
the region
// "bwdistgeodesic" is a MATLAB function that computes the geodesic distance
geodesicDistance = bwdistgeodesic(contour, regionOnContour);
// Define the grown region as the region with distance lower than the
threshold
grownRegion = (geodesicDistance <= numberPixelToGrow);
RETURN grownRegion;
```

An important application for this region growth is to be able to unite pixels that are near each other into a single region. Due to the discretization, some regions of high potential will be broken into multiple but nearby pixels. By using the growth technique that was presented, the pixels will unite to form a solid region on the contour.

C.5.2 Finding Partial contour Orientation

Finding the orientation of a partial contour is crucial to the use of directional magnetic convolutions. The way to do it is to start from the extremity of the partial contour and loop every pixel from this point to find the angle for the next point, as depicted in Algorithm 10-B. Since the image is in a matrix, each pixel has 8 possible neighbors, meaning the angles will always be $\theta = n \cdot \frac{\pi}{4}$, with the value of $n = [1, 2, \dots, 8]$. However, the Algorithm 10-B applies multiple consecutive smoothing on the delta values, meaning the angle will have a lot more than 8 possible values.

Algorithm 10-B. Pseudocode for finding the orientation of a partial contour

```
// pc: Matrix with value 1 on a partial contour, and 0 elsewhere
Function PC_ORIENTATION(pc)
// Make sure the region is thin, with only 2 neighbors everywhere, except
on intersections
pcThin = MorphologicalThinning(pc);

// Remove the intersections from the partial contour, which creates more
partial contours
kernel = [1,1,1; 1,0,1; 1,1,1];
convol2D = convolution2D(pcThin, kernel);
hasLessThan3Neighbours = convol2D < 3;
pcThinNoIntersect = pcThin • hasLessThan3Neighbours;

FOR EACH sub_pc IN pcThinNoIntersect
    // If the sub_pc is open, start from the Extremety
    // Else, choose a random point as the Extremety, and remove one of its
neighbour
    isOpen = any(convol2D < 2)
    IF sub_pc IS isOpen
        isExtremety = convol2D == 1;
        Extremety = chooseRandomPoint(isExtremety);
    ELSE
        Extremety = chooseRandomPoint(sub_pc);
    Sub_pc = removeOnePointNearAfromB(Extremety, sub_pc);
```

```

ENDIF

// Initialize loop parameters
numPixelsRemaining = count(sub_pc > 0);
sub_pcRemaining = sub_pc;
currentPoint = Extremety;
allDeltaX = MatrixOfNAN;
allDeltaY = MatrixOfNAN;

// Loop all the points of the pc from the Extremity
WHILE numPixelsRemaining > 0
    nextPoint      =      findNearestPointFromPointOnPC(currentPoint,
sub_pcRemaining);
    allDeltaX      (currentPoint)      =      XDistanceBetween(currentPoint,
nextPoint);
    allDeltaY      (currentPoint)      =      YDistanceBetween(currentPoint,
nextPoint);
    numPixelsRemaining = count(sub_pc > 0);
ENDWHILE
ENDFOR
allAngles = ATAN2(allDeltaY, allDeltaX);
RETURN allAngles;

```

APPENDIX D GENERAL ELECTROMAGNETISM

D.1 Supplementary nomenclature

This appendix presents the nomenclature that is used exclusively in the following appendices.

L	Length of S
A	Area of the circle S_C
ρ	Radius of the circle S_C
Y_{\max}	Height of the circle S_C
TAC	Total absolute curvature of S
κ	Local curvature of S
x, y	Horizontal and vertical position

D.2 Paths characteristics

D.2.1 Characteristics of the paths between 2 points

This appendix will focus on the desired characteristics of a path that links two points together. Although the trivial path between those points is a simple straight line, the developed technique requires an infinite number of paths to compute the space of probabilities, not only the most optimal one.

For a path between 2 points noted γ_i and γ_f , it is preferable to have a symmetrical path, since it is invariant to the swapping of γ_i and γ_f . Examples of 4 different symmetric paths S_{1-4} are shown at Figure 10-6, with a starting angle of β at points $\gamma_{i,f}$ and a distance of $2 \cdot x_0$.

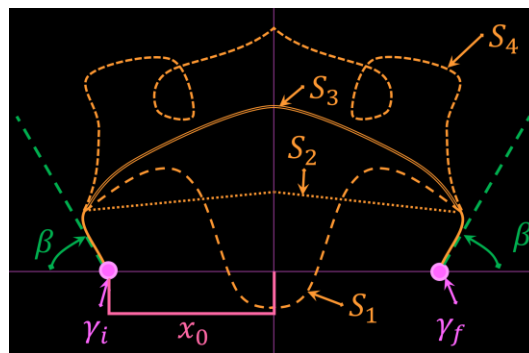


Figure 10-6 : Example of different symmetric paths between points γ_i and γ_f , with a starting angle β

For optimal paths, it is better to have a shorter length L and a smaller total absolute curvature (TAC). The length measures the total distance, as given in equation (135) and the TAC is the integral of the curvature κ in equation (136), with $d\mathbf{s}$ is given at equation (137). For any closed curve, the following inequality is respected $TAC \geq 2\pi$, where it is only equal to 2π for the case of a convex curve.

$$L = \int_S ds \quad (135)$$

$$TAC = \int_S \kappa(\mathbf{s}) d\mathbf{s} , \quad \kappa(\mathbf{s}) = \frac{|\dot{x}\ddot{y} - \dot{y}\ddot{x}|}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \quad (136)$$

$$d\mathbf{s} = \sqrt{\dot{x}^2 + \dot{y}^2} dt \quad (137)$$

Another important characteristic of a path is its smoothness, noted C^k , where k is the number of derivatives of the path that are continuous. The higher is the value of k , the smoother is the path.

With a quick inspection of Figure 10-6, it is easy to determine that an optimal path should not be self-intersecting since it will pass by the same point more than once. Hence, the loop present in S_4 could simply be removed for a shorter path with a lower total absolute curvature. Also, the curve S_1 is concave, meaning that the TAC is not minimized. Finally, the curve S_2 is not smooth since its first derivative is not continuous. Therefore, the only curve in Figure 10-6 that respects all the criteria is S_3 , as seen at Table 3.3.

Table 10.1 : Qualitative Comparison Between the Partial contours Presented in Figure 10-6

Partial contour S_n	Non-self-intersecting	Convex	Smooth
S_1	✓		
S_2	✓	✓	
S_3	✓	✓	✓
S_4			

D.2.2 Choosing the circle, rejecting the parabola

The current appendix will explain why circular paths form optimal sets for this problem, which requires to create paths that pass through 2 points, with 2 defined starting angles β . This gives a

total of 4 conditions on any non-symmetric path, but only 3 conditions on a symmetrical path (since the angle β is symmetric).

If the path is chosen as a polynomial, then there would be an infinite number of possibilities for any polynomial of degree higher than 2 at any angle β . However, the equation (18) requires that there should be a single possible path per angle β , meaning that the only possible polynomial path is a parabola. The problem with parabolas is that the angle β^+ should be greater than $\pi/2$ for a path to exist between γ_i and γ_f , otherwise they would diverge. Also, there is no path in the whole space where $\beta^\pm < \pi/2$, meaning that P_S will be zero, which is not desired.

To solve those problems with the parabolas, we are forced to consider the non-polynomial paths which can respect the given criteria. One of the possibilities is the circle since there is only a single circle that passes through 2 points with a given angle β , it is symmetric, non-self-intersecting, convex, and smooth C^∞ . Also, given 3 points, it is always possible to draw a circle that passes through all of them. If the 3 points are aligned, then it is possible to draw a circle of infinite radius. Therefore, the whole space will be covered, and there will be a circle for every angle $\beta = [0, 2\pi]$.

In summary, the parabola does not fit the required conditions well, while the circle fits them perfectly.

D.2.3 Circular path parameters

The cartesian equation of the circular path S_C is given at (20), with an illustration of all its parameters at Figure 10-7. From this equation, we can easily find the radius ρ given at (138). Also, since the focus is only on the arc $S_C(\beta)$ seen at Figure 4-4, we can define the height between the x axis and the top of S_C as Y_{\max} given at equation (139). Furthermore, the length L of S_C is given by equation (140), and the area A between the x axis and the path S_C is given by equation (141). No proof of these equations is provided since they can be demonstrated with basic trigonometry, and they can be tested for the boundary conditions at $\beta = \{0, \pi\}$ and for the half circle at $\beta = \pi/2$.

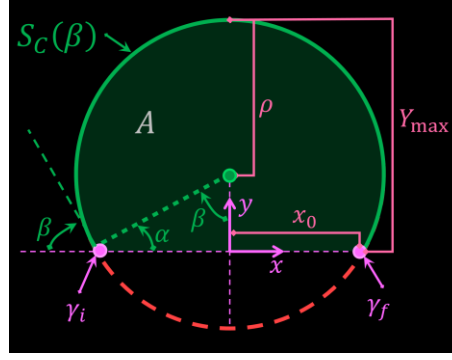


Figure 10-7 : Example of a circular path between points γ_i and γ_f , with a starting angle β

$$\rho = x_0 \csc \beta \quad (138)$$

$$Y_{max} = x_0 \cot \frac{\beta}{2} \quad (139)$$

$$L = \frac{2x_0(\pi - \beta)}{\sin \beta} \quad (140)$$

$$A = x_0^2 \left(\frac{\pi - \beta}{\sin^2 \beta} + \cot \beta \right) \quad (141)$$

D.3 Electromagnetic potential

D.3.1 Elliptical potentials and paths

It was previously discussed with equation (20) that each equipotential curve forms a perfect circle. It is easy to generalize it to any ellipse passing from the same points by using the transformation (142), where b is the semi minor axis. However, this changes the values of the equations (138) to (141), which are outside the scope of the current paper.

Furthermore, such transformations do not obey Gauss law, the conservation of energy, or the diagonal superposition of dipoles of equation (30), meaning that the potential will not be constant inside a closed shape. Hence, they cannot be used for the computation of probabilities. Also, the equipotential lines will not be elliptical, unless the partial contour where the potential is computed is a line, or unless we use the convolution alternative given in appendix “Convolution alternative”.

$$y \rightarrow \frac{y}{b} \quad (142)$$

D.3.2 Convolution alternative

Another way to compute the circular potential without the convolution given in equation (13) is to use the equation (35) directly, with the coordinate system placed at the middle of the line between γ_i and γ_f , and the x axis pointing towards γ_f . Then, using the same definitions of V_m^\pm given at equation (22), we transform the value of V_m with equation (143). This allows to make sure that V_m is positive in the region V_m^+ , and negative otherwise.

Furthermore, using equations (35) along with the transformation (143) might be faster to compute than equation (13) since it does not require the use of convolutions, but it requires additional time to correctly identify the regions V_m^\pm and additional time to process multiple partial contours in the same image individually.

$$V_m \rightarrow \begin{cases} V_m - 2\pi, & V_m^+ \cap (V_m < 0) \\ 2\pi - V_m, & V_m^+ \cap (V_m > 0) \\ V_m, & \text{otherwise} \end{cases} \quad (143)$$

D.3.3 Demonstration that equipotential lines are circular

Starting from V_m given in equation (35) and using definition (38) and identity (37), the goal of this appendix is to demonstrate that the potential V_m has circular equipotential lines. The demonstration is done by finding transforming equation (35) into the parametric equation of the circle given in (40). The full demonstration is given in Figure 10-8 below.

Initial equation

$$V_m = \operatorname{atan}\left(\frac{x+x_0}{y}\right) - \operatorname{atan}\left(\frac{x-x_0}{y}\right)$$

Identity 1: $\operatorname{atan}(X) = \frac{i}{2} \ln\left(\frac{1-ix}{1+ix}\right)$

Identity 2: $\ln(X) - \ln(Y) = \ln\left(\frac{X}{Y}\right)$

$$\Rightarrow V_m = \frac{i}{2} \ln\left[\frac{\left(1 + \frac{i(x+x_0)}{y}\right)\left(1 - \frac{i(x-x_0)}{y}\right)}{\left(1 + \frac{i(x-x_0)}{y}\right)\left(1 - \frac{i(x+x_0)}{y}\right)}\right]$$

Defining $v \equiv e^{-2iV_m}$, $A \equiv \frac{x-x_0}{y}$, $B \equiv \frac{x+x_0}{y}$

$$\Rightarrow v = \frac{(1+Ai)(1-Bi)}{(1+Bi)(1-Ai)} = \frac{1+(A-B)i+AB}{1-(A-B)i+AB}$$

$$A-B = \frac{-2x_0}{y} \quad AB = \frac{x^2-x_0^2}{y^2}$$

$$\Rightarrow v = \frac{y^2(y^2-2x_0yi+x^2-x_0^2)}{y^2(y^2+2x_0yi+x^2-x_0^2)}$$

$$\Rightarrow v(y^2+2x_0yi+x^2-x_0^2) = y^2-2x_0yi+x^2-x_0^2$$

Intermediate form

$$\Rightarrow (v-1)y^2 + (v-1)x^2 + (v+1)2x_0yi - (v-1)x_0^2 = 0$$

$$(v-1)\left(y^2 + 2\frac{v+1}{v-1}ix_0y - \left(\frac{v+1}{v-1}\right)^2x_0^2\right) = -(v-1)x^2 - \frac{(v+1)^2}{v-1}ix_0^2 + (v-1)x_0^2$$

$$\left(y + \frac{v+1}{v-1}x_0i\right)^2 = -x^2 + \left(i\frac{v+1}{v-1}\right)^2x_0^2 + x_0^2$$

$$\left(y + \frac{v+1}{v-1}x_0i\right)^2 + x^2 = \left(i\frac{v+1}{v-1}\right)^2x_0^2 + x_0^2$$

Identity 3: $\cot(X) = -i\frac{e^{-iX} + e^{iX}}{e^{-iX} - e^{iX}}$

$$\Rightarrow \cot(V_m) = -i\frac{e^{-iV_m} + e^{iV_m}}{e^{-iV_m} - e^{iV_m}} = -i\frac{e^{-iV_m}(e^{2iV_m} + 1)}{e^{-iV_m}(e^{2iV_m} - 1)}$$

$$\Rightarrow \cot(V_m) = -i\frac{v+1}{v-1}$$

$$\left(y + x_0\cot(V_m)\right)^2 + x^2 = x_0^2(\cot^2(V_m) + 1)$$

$$\Rightarrow \left(y + x_0\cot(V_m)\right)^2 + x^2 = x_0^2\csc^2(V_m) \quad \square$$

Figure 10-8 : Demonstration that the potential V_m has circular equipotentials