Fast quantitative image processing methods and biological image analysis

Dániel Hillier

A thesis submitted for the degree of
Doctor of Philosophy

Scientific advisor:
Tamás Roska, D.Sc.
ordinary member of the Hungarian Academy of Sciences

Supervisor:
Csaba Rekeczky, Ph.D.

Faculty of Information Technology
Péter Pázmány Catholic University

Budapest, 2007

The truth isn’t flashy.
Flashy words aren’t true.
Educated people aren’t always smart.
Smart people don’t always have an education.
The virtuous do not bandy arguments.
Those who bandy arguments are not virtuous.
The Masters don’t hang on to things.
They’re always doing something for other people,
so they always have more to give.
They give away whatever they have,
so what they have is worth more.
If you want to get right with the Way,
help other people, don’t hurt them.
The Masters always work with people,
never against them.
(Lao-Ce: Tao Te Ching, 81)

Acknowledgements

During the years of my doctoral studies I had the occasion to learn how to do research and quality work from the example of my supervisors. I am grateful to *Tamás Roska* for giving me challenging opportunities, advising in my efforts with empathy and patience.

I am also thankful for *Csaba Rekeczky* for his constant support, for sharing his insights with me and leading me to learn an efficient way of doing research.

I really enjoyed the extremely concentrated, highy interesting research periods spent with *Botond Roska.* I also thank his family for hosting me so many days.

I consider myself very lucky for having the occasion to learn from *Johan A. K. Suykens* who advised me during my year in Leuven. I thank the support of *Joos Vandewalle* during my stay in his lab in Leuven.

I hope I will have another occasion to cooperate with *David Lopez Vilarino*, *Serkan Günel*, *Samuel Xavier de Souza* and *Piotr Dudek.*

I also thank the collaboration and friendship of *Zsolt Czeilinger* and *Viktor Binzberger.*

I thank and enjoyed the company of Gergely Soós, Zsolt Szálka, Eszter Simon, László Orzó, Levente Török, István Petrás, István Szatmári, Dávid Bálya, Gergely Tímár and the whole team of Analogic Computers Ltd.

Help in preparations and discussions related to the classes given in mathematical analysis with Gábor Vásárhelyi, Anna Lázár, Róbert

Wagner, Kristóf Iván, Zsanett Orlovits, Panni Szilvásyné assisted me in developing skills.

I thank *Zsuzsa Vágó* for her selfless devotion in explaining the details of mathematical analysis to me.

I keep a good memory of discussions, soccer matches and other common activities with my fellow doctoral students, namely Gaurav Gandhi, Zoltán Szlávik, András Mozsáry, Zoltán Fodróczi, Csaba Benedek, László Havasi, István Kóbor, Gábor Hodász, Gábor Pohl, György Cserey, Mária Ercsey-Ravasz, Tamás Bárdi and Attila Kis.

I thank the financial and administrative support of the Péter Pázmány Catholic University and the Computer and Automation Research Institute where I spent my PhD years.

The excellent formal and informal teaching work from my undergraduate years also had an impact on my PhD studies. I am grateful for *Gyöngyi Szilágyi*, *father László Kemenes*, father *József Wettstein* and father *András Guba*, *Péterné Bassola*, *Ariadne Bartai* and *Tünde Tőkés*.

Finally I am grateful to *my wife* for her selfless love and thank my *parents* for their generous upbringing, helping me and providing all means for my education and studies. I dedicate this dissertation to them.

Abstract

The cellular neural-nonlinear network (CNN) and the Cellular Wave Computing paradigm was applied to solve a wide range of challenging problems in the last decade. The first problem I have been working on is the real-time three dimensional reconstruction of human heart cavities from ultrasound recordings. Reliable medical diagnosis requires the accurate estimation of cardiac measures like ejection fraction, cavity volume and wall thickening. Currently, medical experts manually measure these indexes prone to subjective bias and error. I developed a computational method providing on-line, automated 3D reconstruction of the right atrium together with the visualization of the atrial septal defect (ASD). I evaluated the accuracy of the method via comparing algorithm results with phantom and clinical echocardiographic data sets manually traced by independent cardiologist experts. Error of the algorithm proved to be comparable to the inter-observer variability between independent experts. Interactive planning of surgical interventions in pediatric cardiology was presented as an illustrative example demonstrating the clinical potential of the method.

The second problem I addressed was the quantification of the dendritic morphology of retinal ganglion cells (RGC). A fundamental paradigm of neuroscience is that neural structure is related to neural function. At the cellular level it was shown in many studies that neurons with different shapes have different function. At a higher organizational level it was shown that the dendrites and axonal arborization of neurons in several brain regions are organized in thin strata and neural function varies across strata. A striking example is the mammalian retina where the inner plexiform layer (IPL) consists of about ten

different, well-defined strata and each strata extracts a different feature from the visual scene forming a stack of image representations in the retina. In brain regions where different functions are encoded in different strata, a powerful morphological approach would be to quantitatively define the strata which are formed by dendritic and axonal arborization of neurons. Then from simply knowing the depth of stratification of a stained neuron one could deduce its function. I developed an automated algorithm that can scan a large number of cells from a retina and quantify the ramification depth of their dendrites.

One candidate mechanism that can play a role in efficient information processing is synchronization. The main problem in analyzing synchronization phenomena in oscillatory networks is that models are non-linear, high dimensional systems. I developed an algorithm that can teach dynamical behavior to a network of oscillators. Using this algorithm, I uncovered two exciting new phenomena. First, I showed examples where switching on coupling between oscillators modified the qualitative behavior of the cells, i.e. proper coupling settings can transform chaotic behavior to periodic or vice versa. Second, I demonstrated for the first time that highly asymmetric interaction weights can synchronize multiple, coexisting cell groups. Despite the asymmetry in the interaction pattern, the spatial layout of the synchronized cell groups followed the topological symmetries of the network.

Contents

Chapter 1

Introduction

The need to increase quality of life and the life expectancy of people induced a remarkable research activity in biology since many decades. Although personalized medicine is the ultimate aim, human expertise will never be sufficient to provide equal, perfect treatement to everyone. The reasons are simple: humans have different expertise and experience and their performance varies in time making fault-less diagnosis and treatment impossible. Research in biology involves a huge amount of manual work. Here too, variations in the performance of humans doing the experiment or the measurement introduces errors in the analysis. Moreover, without proper quantitative methods and statistics, the human brain is prone to misinterpret data.

Both in biology and in medicine, the most fruitful but also the most difficult way of getting insight into mechanisms of living organisms is by simple observation of the subject in its original environment and living conditions. There are relatively few methods that provide information about living organisms without severely interfering with their normal behavior. Non-invasive biological imaging methods are of the most widely applied observation strategies. Both in research and in clinical pratice, X-ray, magnetic resonance and ultrasound based imaging have a history of several decades.

Details of living organisms below around $1mm$ resolution cannot be imaged with these modalities. Before the advent of medical imaging, microscopes were already used widely to observe and analyze functions at low spatial scale. Microscope based methods are almost always invasive and one must be very cautious in infering high level behavioral conclusions from microscope based observations. The gap between low-level and high-level analysis appears to be shrinking due to recent advances in both microscope technology and in molecular biology. Combining high-resolution microscopy with image processing and molecular biology techniques can open the way to functional analysis of mechanisms even at molecular scale.

Morphology of living organisms is extremely variable. Therefore long training and extensive experience is required for human observers to be able to interpret recordings from living organisms.

In the recent years imaging modalities started to evolve into the third spatial dimension. After the first enthusiasm, it became clear that the extra amount of data poses two serious problems. First, data storage requires more resources - this issue falls out of the scope of my work. Second, it is much more difficult to visualize, manipulate and interpret 3D recordings efficiently. Handling 3D data is very difficult to our mind that is traditionally trained on visual information seen in books and on flat screens. The ultimate aim of the analysis is always to make some decisions like "does this image show a healthy heart or not?" or "is this recording showing cell type A or B?". Both questions reveal that quantification has to be performed based on our observations. Stepping into the third spatial dimension provides more information for our decisions but humans actually have less processing ability for three dimensional data.

Our lack of processing ability can be complemented by computers. However, image processing and user interface technologies did not follow innovations in image acquisition technologies. The main reason for this delay is the constant shortage of processing power in computers with respect to the tasks to be solved. The required processing power can be provided by computing architectures based on the Cellular Nonlinear Network (CNN) paradigm ([12, 13, 14, 15, 16, 17]). CNNs are regular, single or multi-layer, parallel processing structures with analog nonlinear computing units (base cells). The state value of the individual processors is continuous in time and their connectivity is local in space. The program of these networks is completely determined by the pattern of the local interactions, the so-called template. The time-evolution of the analog transient, "driven" by the template operator and the processor dynamics, represents the computation in CNN (results can be defined in equlibrium or non-equilibrium states of the network). Completing the base cells of CNN with local sensors, local data memories, arithmetical and logical units, furthermore with global program memories and control units results in the Cellular Nonlinear Network Universal Machine (CNN-UM) architecture [16]. The CNN-UM is an analogic (analog and logic) supercomputer, it is universal in Turing sense [18]. Most algorithms were designed for the two main physical implementations of this architecture: the ACE4k [19] and its successor, the ACE16k [20].

In this dissertation, the focus is put on solving near-sensor reconstruction problems relying on parallel image processing operators. Parallel algorithmic solutions are shown for 3D echocardiography related problems where the demand for an extremely high computing power and the requirement for real-time processing make fully sequential approaches inconvenient. The algorithms were implemented on several processor architectures and it will be shown that the most convincing gain in efficiency occurs when spatio-temporal dynamics of diffusion and wave phenomena are used and executed on a parallel processing array.

The first problem I have been working on is the real-time three dimensional reconstruction of human heart cavities from ultrasound recordings. Reliable medical diagnosis requires the accurate estimation of cardiac measures like ejection fraction, cavity volume and wall thickening. Currently, medical experts manually measure these indexes prone to subjective bias and error. In many cases, analysis of the cavity morphologies is required. In lack of 3D probes and visualization tools, the clinician has to spend considerable amount of time to reconstruct the morphology of the heart in his mind using several 2D views. Reliability of such a reconstruction is questionable due to the high variability of possible morphologies and to the fast moving nature of the heart.

The second problem I addressed was the quantification of the dendritic morphology of retinal ganglion cells. A fundamental paradigm of neuroscience is that neural structure is related to neural function. At the cellular level it was shown in many studies that neurons with different shapes have different function. At a higher organizational level it was shown that the dendrites and axonal arborization of neurons in several brain regions are organized in thin strata and neural function varies across strata. A striking example is the mammalian retina where the Inner Plexiform Layer (IPL) consists of about ten different, well-defined strata and each strata extracts a different feature from the visual scene forming a stack of image representations in the retina.

In brain regions where different function is encoded in different strata a powerful morphological approach would be to quantitatively define the strata which are formed by dendritic and axonal arborization of neurons. Then from simply knowing the depth of stratification of a stained neuron one could deduce its function. I developed an automated algorithm that can scan a large number of cells from a retina and quantify the ramifications depth of their dendrites.

I acquired experience in solving demanding 3D image processing via solving the two above mentioned problems. During my work, I realized that experienced human observers can extract relevant information about 3D structures quickly. The same task is very challenging for computers. I attended neurobiology courses during my doctoral studies and realized that the massively parallel CNN chip I worked on was similar to living neural structures. Discussions with biologists about retinal architecture raised my interest in understanding how neural circuits process information.

One candidate mechanism that can play a role in efficient information processing is synchronization. Synchronization phenomena are apparent in many field of physics and biology but we do not know what synchronization does and how it might play a role in information processing. The main problem in analyzing oscillatory networks is that models are non-linear, high dimensional systems. First of all, the state of the art mathematical toolset can analyse very simplified models only that are quite far from oscillator networks observed in nature. Even for simple models it is very difficult to get closed form expressions on conditions for synchronization.

The desire to explore cooperating behavior in oscillatory networks motivated me to overcome the barrier that our mathematical toolset is very limited to study synchronization phenomena in realistic networks. I developed an algorithm that can teach dynamical behavior to a network of oscillators. Using this algorithm, I uncovered two exciting new phenomena. The first phenomenon is counter intuitive and gave counter examples to existing theory. In [21] Liu et al. claim that when coupling is added to a network of chaotic oscillators with double- or multi-scroll attractor, Lyapunov exponents being zero in the uncoupled system become positive as coupling is increased. They suggest that this rule is general, however, the case when the coupling makes the qualitative behavior of cells simpler indicates that this may not always be true.

The second phenomenon is related to [22] where it was shown that symmetries of the network topology with nearest neighbor connections and uniform interaction weights lead to synchronization in multiple, coexisting cell groups. I demonstrated for the first time, that highly asymmetric interaction weights can also synchronize multiple, coexisting cell groups. In addition, despite the asymmetry in the interaction pattern, the spatial layout of the synchronized cell groups followed the topological symmetries of the network.

Cellular Nonlinear Networks composed of oscillators are high dimensional and non-linear dynamical systems. Developing constructive approaches that can unfold new phenomena is therefore very difficult. My results represent a step toward bridging the gap in our understanding between systems that are simple thus mathematically tractable and systems that are physically more realistic but require large efforts to be analyzed.

The dissertation is organized as follows. Chapter 2 develops Thesis 1, [1] [4] [5] [6] [7] [8], and describes the on-line reconstruction method that extracts the right atrium from 3D ultrasound recordings and quantifies the cavity volumes and the size of the atrial septal defect. In Chapter 3 (Thesis 2) [2] [9] a cell morphology based method is presented that can automatically quantify the ramification depth of ganglion cells in the mammalian retina from 3D confocal microscope scans. Chapter 4 (Thesis 3) [3] [10] [11] explores new forms of cooperative behavior in oscillator arrays. Synchronization is one of the most promising candidate mechanisms that could play a role in efficient information processing in living networks of neurons. Understanding synchronization phenomena could also open a way to design more efficient processor architectures. Chapter 5 summarizes the main results and highlights further potential applications where the contributions of this dissertation could be efficiently exploited.

Chapter 2

On-line 3D echocardiography

2.1 Introduction

Echocardiography is a widely available diagnostic technique allowing direct analysis of the heart in motion in a noninvasive and relatively cheap procedure. For decades, analysis of echocardiograms were evaluated in a qualitative manner where the reliability of diagnoses depended on the level of experience and skill of the cardiologist expert.

Quantitative measurements of classical cardiac indexes - like ejection fraction and the volume of the left ventricle (LV) - were done by tedious manual work prone to subjective bias and error. In addition, in lack of three dimensional (3D) probes and visualization tools, considerable amount of skill and time was needed for the clinician to reconstruct the morphology of the heart in his mind using several 2D views. Reliability of such a reconstruction is questionable due to the fast moving nature of the heart.

In the past decade, many efforts have been made in the medical imaging community to develop automated algorithms in order to assist cardiologists in improving the speed and reliability of daily clinical routine. Although 2D echocardiography is still largely dominant, 3D visualization and quantitative analysis of heart morphology and dynamics have a huge potential in early diagnosis of diseases and in reducing risks of surgical interventions.

A computational method providing on-line, automated 3D reconstruction of the right atrium together with the visualization of the atrial septal defect (ASD) is presented. Massively parallel topographic [1] cellular computational approaches proposed for contour localization and tracking are described. When implemented on a focal plane cellular array microprocessor, these algorithms offer real-time object contour localization and tracking – even at very high frame rates. Three specific methods (Constrained Wave Computing (CWC) continuous time version, CWC discrete time version and Pixel Level Snakes (PLS)) will be described and compared along with their associated hardware-software architectures. Computational complexity, implementation, and performance related issues are discussed on a common platform (ACE-BOX with the ACEx CNN-UM chips). A novel architecture is proposed incorporating the best solutions learned from the comparison.

Endocardial boundaries are extracted from the raw ultrasound data in real-time by topographic cellular active contour (TCAC) algorithms implemented on a massively parallel processor. Extracted boundary point sets corresponding to the endocardium are rendered into a 3D mesh enabling the quantification of the volume of the right atrium and the visualization of the ASD.

Accuracy of the boundary tracking was assessed by comparison with phantom and clinical echocardiographic data sets manually traced by independent cardiologist experts. Accuracy of volume quantification was validated on 6 in-vitro static phantoms and 6 clinical volumes. A new aspect of the validation is the application of nonlinear wave metric for contour comparison. For selected clinical examples, error of the algorithms proved to be comparable to the inter-observer variability between independent experts. Interactive planning of surgical interventions in pediatric cardiology is presented as an illustrative example that demonstrates the clinical potential of the method.

I describe an experimental study and the underlying automated, high speed computational method that is able to extract the right atrium and the ASD from 3D ultrasound data. The method provides quantitative measurements in order to assist diagnosis and 3D on-line virtual surgery planning. Fig. 2.1 shows the main processing steps from data acquisition to the virtual surgery planning scene.

[1] The notion "topographic" is used in our context as the description of a structured entity showing and preserving the relations among its components. To be specific, the massively parallel hardware we used is topographic with respect to the 2D data flowing from the ultrasound sensor.

3D data was acquired at 720 × 512 pixels and 90 frames per volume resolution using a Philips Sonos 5500 with fast rotating transducer. Endocardial boundary detection was performed on the right and left atria using a real-time topographic contour tracking algorithm running at $25-500$ frames per second (fps) depending on the hardware used for implementation. Endocardial boundary detection and high resolution mesh construction of the right and left atrium was done in one minute.

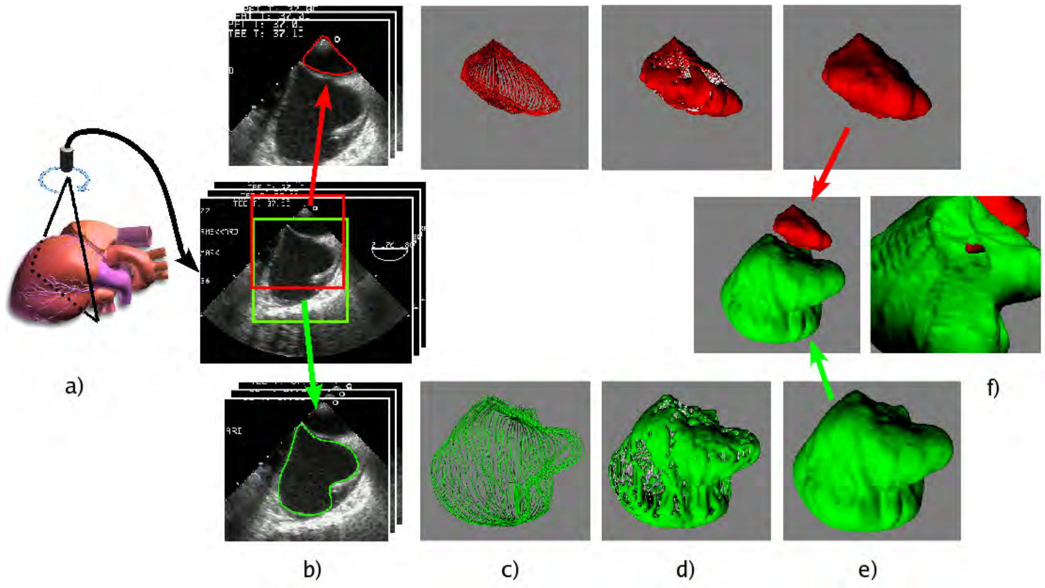


Figure 2.1: Processing steps from ultrasound data acquisition to the interactive virtual surgery planning scene. a) Full resolution images are acquired using a fast rotating transducer. b) Real-time TCAC method extracts the endocardial boundaries from the 128 × 128 pixels resolution subflows. Next, and a smoothing spline is fitted to each sampled contour to eliminate noise and outlier points. c) Contours are rotated and translated into their appropriate 3D position d) A mesh is constructed from each of the equally spaced point clouds using an implicit surface rendering (metaball) technique. e) The reconstructed atria are merged, the volume of the RA and the size of the ASD is calculated f) In a virtual reality scene, the model of the atria can be opened from a desired viewpoint so that the morphology can be inspected. Observe the atrial septal defect appearing as a red spot. Heart model: courtesy to zygote.com

To avoid confusion, the terms ”3D real-time” and ”3D on-line” are clarified. On commercial echo machines, volumes acquired with a 3D transducer are displayed right after data is collected, i.e. data acquisition and voxel based display is performed in "3D real-time". In contrast, quantitative diagnosis requires a geometrical model based reconstruction of the examined cavity that requires additional computation time and is currently done semi-automatically for the LV on state-of-the-art systems. I introduce an approach that provides the clinician quantitative measurements on-line, i.e. in at most 1 minute after data acquisition.

2.2 Related works

A prerequisite for quantitative analysis and model based 3D reconstruction is the accurate segmentation of the raw ultrasound (US) data. This is a notoriously difficult task that has been addressed by many research groups. Related studies were so far focusing on the LV. Recently, work on data sets acquired using a matrix array transducer referred to as real-time 3D (RT3D) probe was reported, see [23] for a review. A review of the whole field of cardiac modeling can be found in [24].

Early approaches aiming to solve the boundary segmentation problem of 3D echocardiographic images included approaches based on mathematical morphology [25], Markov random fields [26], fuzzy logic [27], neural networks [28] and active contours [29]. In recent years some more complete studies have been published. In [30] Bosch et al. introduced a fully automated segmentation method of the LV based on active appearance models (AAM). Their algorithm, validated on a large number of unselected clinical examples, showed high accuracy. Mitchell et al. [31] extended the AAM approach into 3D and applied it to 3D magnetic resonance imaging (MRI) and 2D+T (time sequence of 2D slices from the same view) echocardiographic data. In [32] Beichel et al. extended the active appearance method to cope with gross disturbances caused by pathological cases.

Corsi et al in [33] used a level-set based method applied directly in 3D space to segment the LV from real-time 3D acquisitions. Paragios [34] constructed a level-set framework combining a priori spatio-temporal shape knowledge and local variability information to track the boundary of the LV. Yang et al [35] applied level-sets combined with interobject constraints to segment medical images in a multi-object, maximum a posteriori estimation framework.

Wolf et al. in [36] presented a system that extracts points belonging to endocardium using a multi-scale edge criterion. These points are first locally then globally connected via optimal path search yielding to a closed contour. Abolmaesumi et al. [37] introduced an algorithm to interactively segment cavities from US images using multimodel probabilistic data association filters. Song et al. [38] used a Bayesian framework to infer a correct surface model of the LV adapted to both image derived information and high-level a priori shape knowledge.

Rekeczky et al. [39] used optimal non-linear filtering combined with wave computing [40] to detect the endocardium of the LV in real-time. Montagnat et al. [41] used anisotropic diffusion as preprocessing filter to fit a deformable surface to image data. Later [42], they used 3D deformable models with constrained deformations to segment cylindrical echocardiograms. In [43] Angelini et al. used 4D directional space-frequency analysis to reduce speckle noise and to enhance anatomical structures on RT3D recordings. Deformable model based segmentation performed on the denoised data in 2D was used to extract the endocardial boundary of the LV.

Gérard et al. [44] fitted a discrete closed surface model of the LV to image gradient derived information constrained by a simple biomechanical motion model they obtained from MRI recordings. Their approach worked for both 2D slices rotated into 3D and for RT3D data, however the spatial resolution was low to keep processing time within an acceptable boundary. Zagrodsky et al. [45] presented a similar but fully automated approach for RT3D data segmentation and analysis.

Jacob et al. [46] developed a semi-automatic method to track the myocardial boundaries of the LV using a fusion of snakes and Kalman filters. In their work, the borders are decomposed into regional parameters to quantitatively assess wall movements and myocardial thickening. Sanchez-Ortiz et al. in [47] combined phase-based local boundary detection with global image information extracted by multiscale fuzzy clustering. The segmentation result is fitted to the spatiotemporal model of the LV. Their approach is fully automatic and it can be adapted to RT3D data but the algorithm is computationally expensive. Comaniciu et al. in [48] presented a robust, real-time information fusion framework to track the myocardium of the LV. Applying strongly adapted principal component analysis model that incorporates information from system dynamics, measurement noise, subspace shape model and from the initial contour they show outstanding results with high processing speed.

For the right atrium, very few computer aided tools were proposed. In [49] multiple cardiac chambers including the right ventricle of the fetal heart were segmented from a 3D US volume. Their method improved on the level-set based approach of [50] by introducing a collision detection term to avoid adjacent chambers be invaded by the neighbor snake.

Regarding related works summarized above, I contributed in

- providing the first study on the geometry based 3D reconstruction of the right atrium and the ASD. Compared to the LV, this task is more difficult since atria have more complex geometric properties. Contrary to LV studies, simple geometric assumptions do not hold that motivated the use of implicit methods.
- the application of very fast boundary tracking methods implemented on a massively parallel processor.
- measuring the volume of the right atrium (RA), the size of the ASD, and providing support for planning surgical interventions.
- application of a new type of metric when comparing contours for algorithm control and validation.

2.3 Theory of topographic cellular active contour (TCAC) methods

Active contours have become a popular tool in multiple image processing tasks like segmentation, tracking and modeling during the last decade. An active contour (AC) is a parametric curve $u(s) = [x(s), y(s)]^T, s \in [0, 1]$ with $u(0) = u(1)$ and $u'(0) = u'(1)$, evolving from its initial shape and position under the influence of internal and external forces. The AC evolves until it reaches a local minimum of the total energy:

$$E(u) = \int_0^1 \left[\alpha \cdot \left| \frac{\partial u(s)}{\partial s} \right|^2 + \beta \cdot \left| \frac{\partial^2 u(s)}{\partial s^2} \right|^2 + P_{control}(u(s)) \right] ds \qquad (2.1)$$

where the first two terms of the right-hand side represent the internal energy, i.e. the first factor is the membrane energy incorporating the resistance of the AC to stretching, the second is the thin-plate energy accounting for the resistance to bending. The internal energy is balanced with the energy defined by an external potential field ($P_{control}$) derived from the image features. Usual terms may include the gradient of the image convolved with a Gaussian in order to reduce noise $\gamma \cdot |\nabla(G_\sigma(u(s)) * I(u(s)))|$, the image intensity $\zeta \cdot I(u(s))$ or the distance to the nearest boundary (target) point obtained by an edge detector $\eta \cdot \exp^{-d(u(s))^2}$. Constants $\alpha, \beta, \gamma, \zeta, \eta$ balance the influence of internal and external force terms and are chosen depending on the particular application.

An admissible contour function $u(s)$ minimizing Eq. (2.1) must satisfy Euler's differential equation [51]:

$$-\frac{\partial}{\partial s}\left(\alpha \cdot \frac{\partial u(s)}{\partial s}\right) + \frac{\partial^2}{\partial s^2}\left(\beta \cdot \frac{\partial^2 u(s)}{\partial s^2}\right) + \nabla P_{Control}(u(s)) = 0 \tag{2.2}$$

In order to numerically compute a minimal energy solution, the curve $u(s)$ is approximated with linear combinations of basis functions. The most used methods are representations based on local-support basis functions like the finite differences method [52], the finite elements method [53] and the B-splines [54]. Minimization of the functional $E(u)$ requires high computational efforts that limits the application of traditional AC techniques in real-time applications.

Image segmentation can also be formulated in implicit models. Motivated by [55], [56], [40] a propagating active contour can be defined by a geometric flow. The propagation velocity is made up of two terms, the regularity of the contour and image derived information. The model of CWC is given by a reaction-diffusion type partial differential equation (PDE):

$$\frac{\partial I_P(x,y,t)}{\partial t} = div\ grad(I_P(x,y,t)) + \mathcal{F}_1(I_{Control}(x,y,t_0)) + \mathcal{F}_2(I_P(x,y,t)) \tag{2.3}$$

where $\mathcal{F}_1$ and $\mathcal{F}_2$ are nonlinear functions, $I(x,y,t) : [0,N]^2 \times [0,T] \to [0,M]$.

The solution of the PDE can be approximated using spatial discretization made in equidistant steps in both directions, $\Delta x = \Delta y = h$. This way, $I_P(x,y,t)$ is mapped onto a CNN array such that the state value of a CNN cell at a grid point i,j is associated with $I_P(ih,jh,t)$. Using Taylor-series expansion of $I_P(x,y,t)$ the

CNN template corresponding to the second spatial derivative can be obtained [57]. The image derived spatial constraint term $\mathcal{F}_1(.)$ is replaced by the linear combination of the values stored in $I_{Control}(i,j,t_0)$ and $\mathcal{F}_2(.)$ is replaced by the nonlinear function g. The ordinary differential equation approximating the solution of Eq. (2.3) at the grid point i,j can be formulated as:

$$\begin{aligned}\frac{dI_X(i,j,t)}{dt} =& -I_X(i,j,t) + g(I_P(i,j,t)) + \frac{c_1}{4}[I_P(i-1,j,t)+ \\ &+ I_P(i+1,j,t) + I_P(i,j-1,t) + I_P(i,j+1,t)] + z(i,j) \qquad (2.4)\end{aligned}$$

where $I_X(i,j,t)$ represents the state variable at each grid point; $I_P(i,j,t) = f(I_X(i,j,t))$ and $g(.) = c_0 \cdot f(.)$; the nonlinear, sigmoid-type function f is defined as $f(I_X(i,j,t)) = 0.5 \cdot (|I_X(i,j,t)+1| - |I_X(i,j,t)-1|)$. Velocity of the propagating front is depending on the actual content of the image formulated in the term $z(i,j) = z_{const} + \sum_{k,l \in S_1(i,j)} b_{(k,l)} I_{Control}(k,l,t_0)$ where $S_1(i,j)$ represents the 3×3 neighborhood of the cell i,j, contributions of the neighbors are weighted by the values of $b_{(k,l)}$.

Compared to existing AC techniques (see e.g. [52] [29]) TCAC methods discretize the contours at the same order as the spatial variable in the image space and performs parallel computational operations on each contour pixel. The contour is represented as sets of eight-connected black pixels in a binary image I_P. This binary image has the same dimensions as the original image containing the objects or regions to be detected.

2.4 Implementation of topographic cellular active contour (TCAC) methods

To avoid confusion, note that contrary to classical image processing conventions, in our CNN based approach, darker pixels have higher grayscale intensity value. Input data is a discretized, grayscale image $I_{Control}(i,j,k)$: $[1,M] \times [1,N] \times [1, Frnum]$ taken from an US video flow (or "3D data cube") at the k^{th} frame. This grayscale image contains the image features that represent the region to be detected. The algorithms provide the result either as a black contour ($I_C(i,j,k)$), or as a black patch ($I_P(i,j,k)$) on white background. In the following, spatial

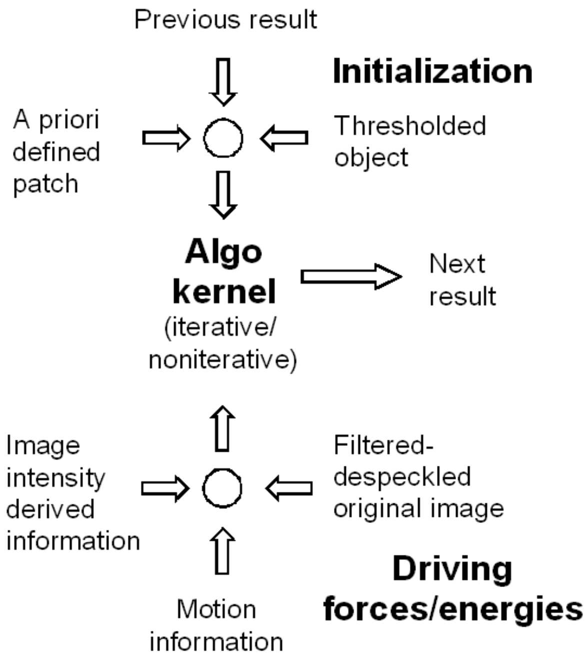


Figure 2.2: The algorithm kernel computes the evolution of curve pixels under the influence of internal and external driving forces. The initial contour can be obtained by either defining it a priori or by extracting an initial feature map obtained from the input image.

index i and j and temporal index t will many times be omitted for better readability.

The evolution of discrete curve points can be implemented either by iterative or by dynamic methods (see Fig. 2.2).

The main difficulties to be overcome reside in the initialization of the contour tracking and the handling of false or missing features on the input image. Initialization is a twofold problem. At the beginning of the contour tracking process the algorithm has to find the approximate position of the target object in the input image - this will be referred to as "system-level calibration". Then the parameters of the tracking algorithm have to be tuned to actual image characteristics like changing illumination level - this will be referred to as "sensor-level calibration". Once the contour has been initialized the quality and speed of subsequent contour detections can be substantially improved by exploiting the correlation between

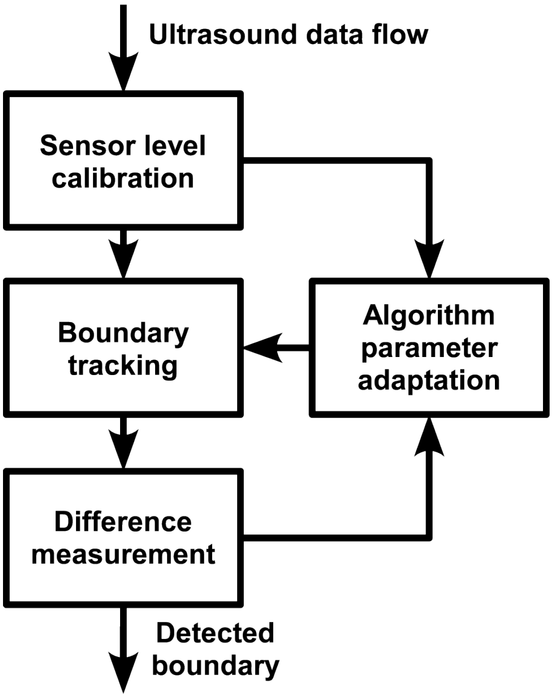


Figure 2.3: System-level view of TCAC techniques. Contour localization is performed together with calibration, error measurement and parameter tuning. Note that ”sensor-level” calibration might not be performed in each iteration.

consecutive frames. Topographic active contour techniques make use of the result from the previous iteration which is used to initialize the current contour detection. This concept is illustrated in Fig. 2.3.

2.4.1 Preprocessing and Calibration

In the initial stage, preprocessing may be needed to eliminate speckle noise from the input images. Various filter algorithms (mode, mean, median, truncated median, globally and locally adaptive truncated median) were implemented on digital platforms (personal computer (PC) and digital signal processor (DSP)) and their performance were evaluated using signal to noise ratio (SNR) histograms and diagrams. Our results showed that the highest SNR increase can indeed be achieved by the mode filter, in accordance with ultrasound speckle models, (see for example [58] and [37]), but it has its payoff in destroying many delicate image features, most notably the thin segments of cardiac walls.

The mean filter is functionally similar to the simple diffusion operator implemented on the CNN architecture. The SNR enhancement is comparable to the result of the mode filter, but the feature distortion is much less severe. Since the implementation is readily available on the CNN platform, using only 160ns time - equal to six convolutions with Gaussian kernel - on the ACE16k chip, the current TCAC implementations can use this method of filtering as a preprocessor. The best image improvement results were achieved by the various truncated median methods. The averaged SNR improvement is less than that of the mode and mean filters, but the feature conservation is much superior. This has a price in implementation complexity, which currently does not make a hardware supported CNN implementation possible, and this in turn results in high computational cost.

Two kinds of calibration modules are implemented. The role of the system-level calibration module is to keep the target object in focus with optimal scaling settings to fit the given size of our topographic computational platform (e.g. 64×64 cells in the case of the ACE4k architecture or 128×128 cells for the ACE16k). We have also made efforts to develop a fully automated way to detect the position of the target object on the first frame of processing, but the results are not yet satisfactory. The system-level calibration module still requires a very simple input from the user, who has to position a window on the target object at the beginning of processing.

The sensor-level calibration module provides the algorithm with initial parameters adapted to the input image under processing. It cuts out 1D samples from the image in radial directions, and uses a Sobel-like 1D edge detection operator (1D version of the Dir_Gradient operator described in the Appendix) on each to find the point where the sampling line intersects the cardiac contour. For more robustness, the sample pixels are calculated from the input image using a 9-neighborhood convolution mask which weights the neighbors lying orthogonally to the sampling line with greater weight. That is basically an averaging parallel to the supposed cardiac wall. The resulting approximate cardiac contour points are checked against some confidence criteria and the admitted points are fed into a L_2-norm ellipse-fitting algorithm (Fig. 2.4). This whole algorithm takes only 20 ms on the Texas Instruments C6020 DSP and outputs the approximate position and size of the target object with the average illumination level of the cardiac chamber and wall.

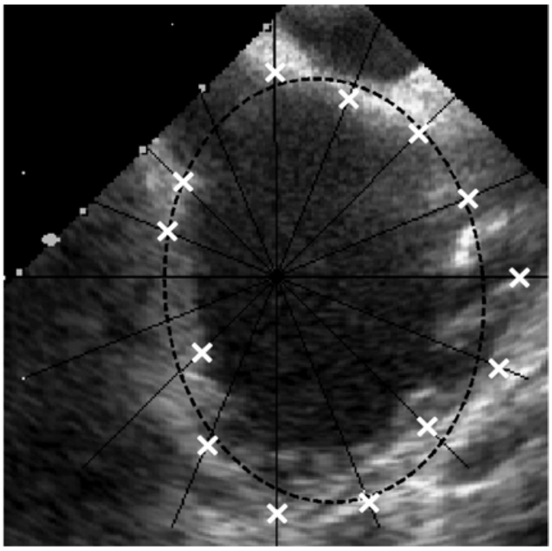


Figure 2.4: Sensor-level calibration provides the algorithm with initial parameters adapted to the input image under processing. 1D samples from the image are extracted in radial directions. A Sobel-like 1D edge detection operator is applied to each sample to find the point where the sampling line intersects the cardiac contour. The resulting approximate cardiac contour points (marked with white X) are checked against some confidence criteria and the admitted points are fed into a L_2-norm ellipse-fitting algorithm.

The sensor-level calibration module can be run for each frame before the main algorithm. This scenario follows accurately the changing illumination and size parameters of the chamber to be tracked at the expense of using 20 ms processing time for each frame. In a different scenario, the sensor-level calibration can only be run for the first frame. In this case, if the difference measuring module (see Fig. 2.3) signals the violation of a priori shape and motion assumptions, the algorithm will be re-initialized by re-running the sensor-level calibration module. This scenario saves processing time by running the calibration module only when really needed. For more details on preprocessing and calibration see [59].

2.4.2 Massively Parallel Processor Architectures

Low-level image processing operators like filtering, edge detection, binary hole filling, feature extraction, etc. are computationally intensive. These operations are inherently pixel-parallel, i.e. identical, localized operations are performed on every pixel. Efficient image processing systems can be designed by associating each image pixel with an image processing circuitry and allowing local connections between neighboring processing cells (Fig. 2.5). Each cell can have local memories and can perform basic arithmetic and logic operations on pixel values of their local neighborhood. CNN [60] represent a powerful framework for this concept. In many CNN implementations, each individual cell circuitry is a realisation of Eq. (2.4), i.e. CNNs can be used to approximate solutions of PDEs. A number of different CNN processor implementations are available for parallel image processing [61] [62] on which various difficult image processing problems were solved at high speed [63].

2.4.3 Constrained Wave Computing

CWC (see preliminary work in [40]) is a TCAC method that can extract endocardial boundaries from a series of discretized, grayscale images taken from an US 3D data set at a specific time instance at high speed. The only a priori information used is that the contour evolution is initialized inside the target object and the boundary to be detected is composed of locally bright regions with possible holes. The contour to be detected is the steady contour of a dynamic wave initiated from patches called sources. The evolution of the dynamic wave can be stopped using a grayscale spatial constraint calculated from the input image(s). Where wall segments are missing on the US image, proper spatial constraint or external force cannot be generated to stop the wave propagation. However, a properly chosen time-constraint can always be applied and thus a solution will be obtained in a non-equilibrium state of the network. In summary, in the case of CWC the task to be solved should be converted into adequate spatio-temporal constraints.

US signal attenuation and dropout make endocardial boundary detection a very challenging problem. Near inter-cavity holes or near wall segments missing

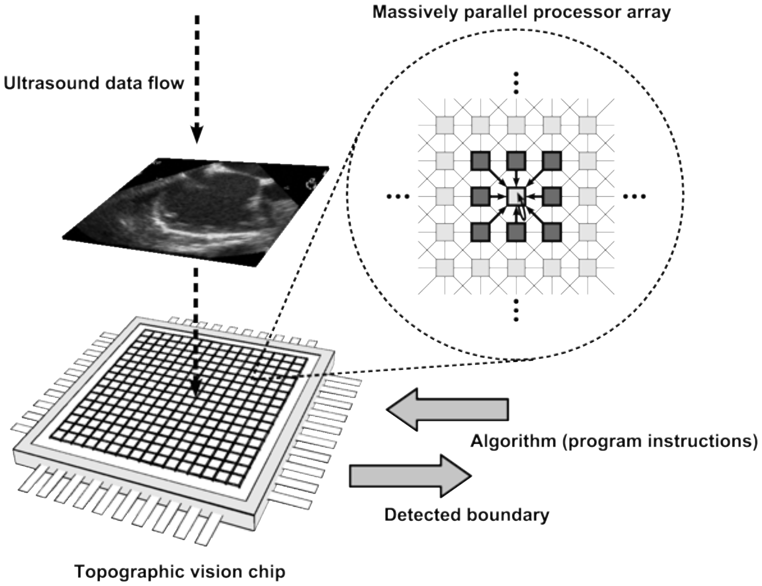

Massively parallel processor array
Ultrasound data flow
Algorithm (program instructions)
Detected boundary
Topographic vision chip

Figure 2.5: Computational framework for pixel-parallel topographic active contour algorithm. Ultrasound data flowing from the transducer is mapped onto a processor array. Each cell on the array has local connections to its neighbors and contains local memories. A cell computes its current state value by performing basic arithmetic and logic operations on pixel values of their local neighborhood and its own previous state value.

due to signal dropout, nearby parts of the AC could lock onto false image features located outside the cardiac wall. Traditional AC approaches try to overcome such problems via embedding some kind of higher knowledge into Eq. (2.1) in order to find $u(s)$ corresponding to the desired minimum of Eq. (2.1). TCAC methods achieve high speed detection of the inner boundary of the cardiac cavity by pixel-parallel algorithms using only local operations. Operations working on the 3×3 neighborhood detect edges locally and thus cannot be aware of whether that edge corresponds to the inner or to the outer side of the cardiac wall.

Higher level knowledge of ”approaching edges from inside” is embedded into CWC to minimize the attraction force of false features in the raw grayscale image $I_{Control}(i, j, t)$. Deforming a filled patch instead of a one pixel wide contour to detect the target region makes the inner and outer side of the contour unambiguous even if only operations working on the local neighborhood of the pixels are used.

In CWC a filled black "patch" $I_P(i,j,t)$ in a binary image is propagating controlled by the raw ultrasound data $I_{Control}(i,j,t)$ to fit the cardiac cavity to be segmented. The functions of internal and control energy terms of Eq. (2.1) are represented as constraints applied to the local neighborhood of each pixel. Internal constraints ensure the proper shape of the patch whereas control constraints ensure that the boundary pixels of the patch in $I_P(i,j,t)$ correspond to the locally bright regions in $I_{Control}(i,j,t)$.

Choosing constants $c_0 = 3$ and $c_1 = 1$ and $b_{0,0} = -2$ and $z_{const} = 3.75$ in Eq. (2.4) a propagating binary wave process is generated. When $I_{Control}$ is the raw ultrasound data taken from the input data set at a specific time instant and I_P is a small black patch inside the cavity, this constrained wave process can extract the endocardium from the raw ultrasound data in a single instruction on [20].

Using explicit Euler discretization with $h = 1$, Eq. (2.4) can also be solved in the discrete time domain. In that case Eq. (2.4) becomes:

$$\begin{aligned} I_X(i,j,t+1) =& g[I_X(i,j,t)] - I_P(i,j,t) + \frac{c_1}{4}[I_P(i-1,j,t)+ \\ &+ I_P(i+1,j,t) + I_P(i,j-1,t) + I_P(i,j+1,t)] + z(i,j) \end{aligned} \tag{2.5}$$

where $I_P(i,j,t) = f[I_X(i,j,t)]$; $g[.]$ and $f[.]$ remain as defined in Eq. 2.4. The result contours obtained via iterating Eq. (2.5) in time on the same processor [20] gives the same result as in the continuous time case. The discrete time solution is obviously much slower but it can be solved on a much wider range of processors. To achieve fast computing time, the discrete time model was decomposed into a series of operations that exploit digital processor architectures better. The performance gain is achieved by using simple, pixel parallel, rule-based operators and perform costly grayscale calculations only at the local neighborhood of the contour. These optimizations were added during the development of the Moving Path Method by Viktor Binzberger.

The contour pixels evolve under the influence of local statistics computations and morphological operations in order to obtain well-defined contours fitting to the endocardial boundaries. For every pixel in $I_P(i,j,t)$, constraints are checked in a properly defined sequence to find the new value of $I_P(i,j,t)$, i.e. to fit $I_P(i,j,t)$ to image features in $I_{Control}(i,j,t)$ corresponding to the endocardium (see Fig. 2.6). The internal constraints can be represented by hit-and-miss masks. Hit-and-miss is a general binary morphological operation that can be used to look for particular masks (patterns) of foreground and background pixels in an image. Internal constraints ensure that the patch will not contain singular white pixels, and fill up the deeper concavities along the boundary of the patch. ”Must be black” mask checking (see Fig. 2.6) sets pixels of the patch image black that have a neighborhood of more than four black pixels. ”Valid black” masks select pixels having exactly three neighboring black pixels that are adjacent to each other. The selected pixels correspond to the edge of the patch that is deformed under the guidance of control constraints to fit image features corresponding to a cardiac cavity. Pixels in $I_P(i,j)$ selected neither by the ”must be black” nor by ”valid black” masks are turned to white eliminating singular black pixels disconnected from the main patch.

Control constraints are applied only to the result of internal constraint checking, i.e. to the current edge pixels of $I_P(i,j,t)$. Two types of external constraints are applied to $I_{Control}(i,j,t)$: global thresholds and a threshold imposed on intensity differences in the local neighborhood. Pixels of $I_{Control}(i,j,t)$ corresponding to pixels of $I_P(i,j,t)$ that were selected in the internal constraint checking phase are checked whether they fall below or above a lower and higher global threshold level (LowThresh, StopThresh), and they are set to black or to white respectively. These thresholds are responsible for the unconditional expansion or shrinking of the contour in regions belonging unequivocally to chambers or myocardium.

For the remaining pixels, the algorithm calculates the average grayscale level of the neighboring pixels on the control image corresponding to the black and white pixels in the patch image, respectively. If the difference between these ”white” and ”black” average values is below a local difference threshold, it sets the pixel to white, otherwise to black.

This last step is the essential contour detection step. By computing the intensity difference between the inner and outer side of the contour, the algorithm approximates the component of the local gradient orthogonal to the boundary of the patch. The patch is expanding if the intensity gradient is greater than a local difference threshold, and shrinking when it is less. In other words, if the cardiac wall is approached from the inside a gradual increase can be seen in the average

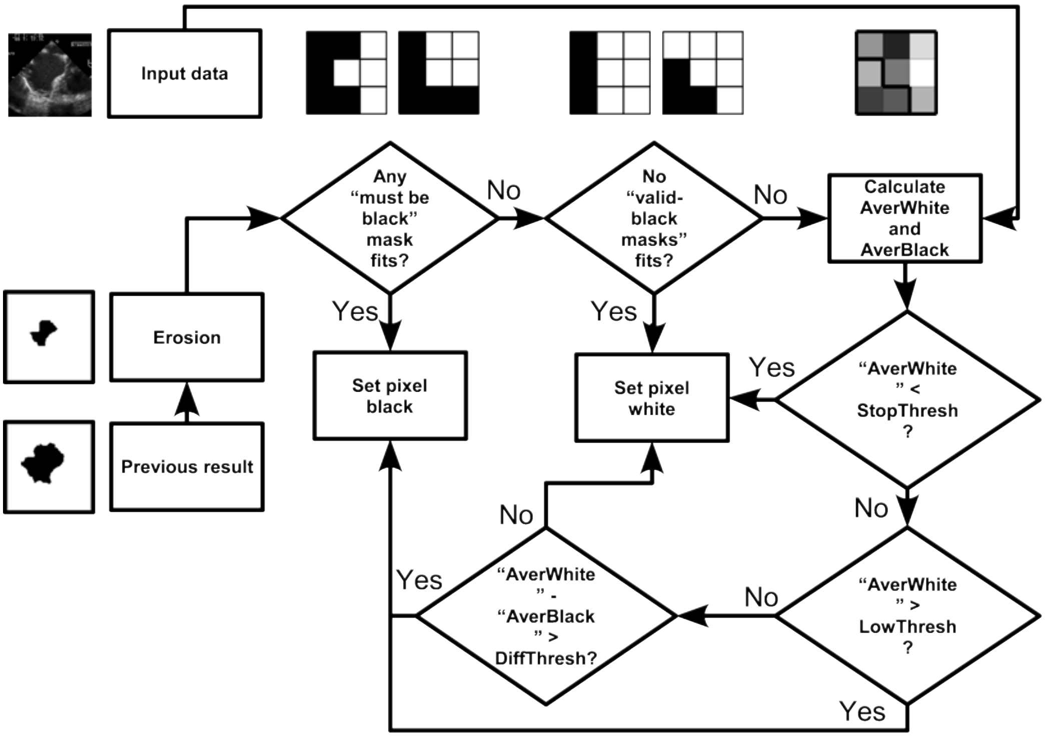

Input data
Any "must be black" mask fits?
No
No "valid-black masks" fits?
No
Calculate AverWhite and AverBlack
Erosion
Yes
Yes
Set pixel black
Set pixel white
Yes
"AverWhite" < StopThresh ?
Previous result
No
No
Yes
"AverWhite" - "AverBlack" > DiffThresh?
No
"AverWhite" > LowThresh ?
Yes

Figure 2.6: Flowchart of the real-time boundary tracking algorithm. On all pixels of the eroded version of the result detected in the previous frame, "must be black" masks (rotated masks with 90 180 and 270 are not shown) sets pixels of the patch image black that have a neighborhood of more than four black pixels. "Valid black" masks (rotated versions not shown) select pixels having exactly three neighboring black pixels that are adjacent to each other. The selected pixels correspond to the edge of the patch that is deformed under the guidance of control constraints to fit image features corresponding to a cardiac cavity. Pixels in $I_P(i,j)$ selected neither by the "must be black" nor by "valid black" masks are turned to white eliminating singular black pixels disconnected from the main patch. External constraints are applied to the grayscale ultrasound image in form of global thresholds and a threshold imposed on intensity differences of the local neighborhood of a pixel. The pixels of the grayscale image corresponding to pixels of the patch image selected in the internal constraint checking phase are checked whether they fall below or above a lower and higher global threshold level (LowThresh, StopThresh), and they are set to black or to white respectively. These thresholds are responsible for the unconditional expansion or shrinking of the contour in regions belonging unequivocally to chambers or myocardium. For the remaining edge pixels of the patch image, the average grayscale level of the neighboring pixels on the control image corresponding to the black and white pixels in the patch image are calculated. If the difference between the "white" and "black" average (AverWhite, AverBlack) values is below a local difference threshold, the pixel is set to white, otherwise to black.

intensity until the intensity plateau of the cardiac wall is reached. This change of gradient amplitude is detected by this method.

In summary, expansion and shrinkage of the patch is coarsely regulated via fast global threshold checking. Computationally more expensive, fine regulation calculates the orthogonal component of the local gradient.

The pseudo-code description of the CWC algorithm is listed in Algorithm 1.

Algorithm 1 Pseudocode of the CWC method in the recursive state

```
function $I_{CWC}(I_P(k-1), I_{Control}(k), err(k-1), \tau)$
    $[w, td, tc] = TuneParams(err(k-1))$
    $I_{SF}(k) = SpeckleFilt(I_{Control}(k))$
    $I_{MIM}(k) = MotionEst(I_{SF}(k+1), I_{SF}(k-1), \tau)$
    $I_{Ext}(k) = Combine(I_{MIM}(k), I_{Control}(k), w)$
    $I_{BWMask}(k) = CalcBWMask(I_P(k-1), I_{Fixmask}, t_d)$
    $I_{Kernel}(k) = TrW_{Erosion}(I_P(k-1), t_d)$
    $I_Y(k) = TrW_{Constr_Dilation}(I_{Ext}(k), I_{BWMask}(k), I_{Kernel}(k), t_c)$
    $err(k) = d_W(I_P(k), I_P(k-1))$
end function
```

Both for the initial frame and in the recursive processing phase the spatial constraints are calculated in the following way. The slightly diffused version of the raw image and the motion intensity map, $I_{MIM}(k)$ are generated from the filtered images. These images reflect spatial information about the intensity around the boundaries and the boundary motion. Motion intensity map is estimated from the difference of the previous and the current frame. More robust motion information could be estimated, if we added the difference of the next frame and the current frame to the difference of the current and the previous frame. If this is the case, we have to read a frame ahead.

These maps are normalized and then combined to form the external driving forces for the wave propagation. This image will be the grayscale spatial constraint that will stop the contour evolution.

Hard limit for wave propagation can be imposed by exploiting a priori information concerning the maximum boundary displacement (t_d) between consecutive frames. Inflating $I_P(k)$ using a wave-type operation with time t_d defines a black and white (BW) mask image to limit the contour evolution. This mask image is

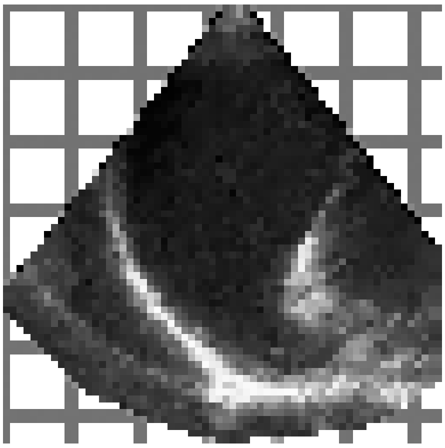


Figure 2.7: A sample, square-sized ultrasound image. The region that falls outside the ultrasound beam is marked by the square pattern.

added to another mask image that disables computation on cells that fall outside of the span of the ultrasound beam (see Fig. 2.7).

In the continuous time version of CWC, the evolution of the active contour is expanding by nature. However the kernel or source for CWC is obtained by deflating $I_P(k)$ with t_d time constant, thus image features moving inwards can also be tracked. Time constant t_d is currently a manual parameter but could also be estimated during calibration. Note that maximum boundary displacement is the same inwards and outwards from the previous result thus this time constant is the same for inflation and deflation. The actual contour evolution is performed by the TrW_Constr_Dilation operation (TrW stands for Trigger Wave). Time constant t_c is the same as t_d during recursive processing, but for the first frame it has to be be set by the sensor-level calibration module.

In the recursive state, the CWC algorithm first evaluates the error from the previous iteration - this is performed in the $TuneParams$ module - then compares it to a cumulated error measure and tunes the weights w and the time parameters t_d and t_c. If the shape or size change is higher than the a priori defined value then a reinitialization will occur. In such a case the recursive processing will be interrupted and re-calibration will be performed. In our framework we used both Hamming-type metric and Hausdorff-type metric for error estimation but non-linear Wave-type metric proved to be the most reliable and robust. This

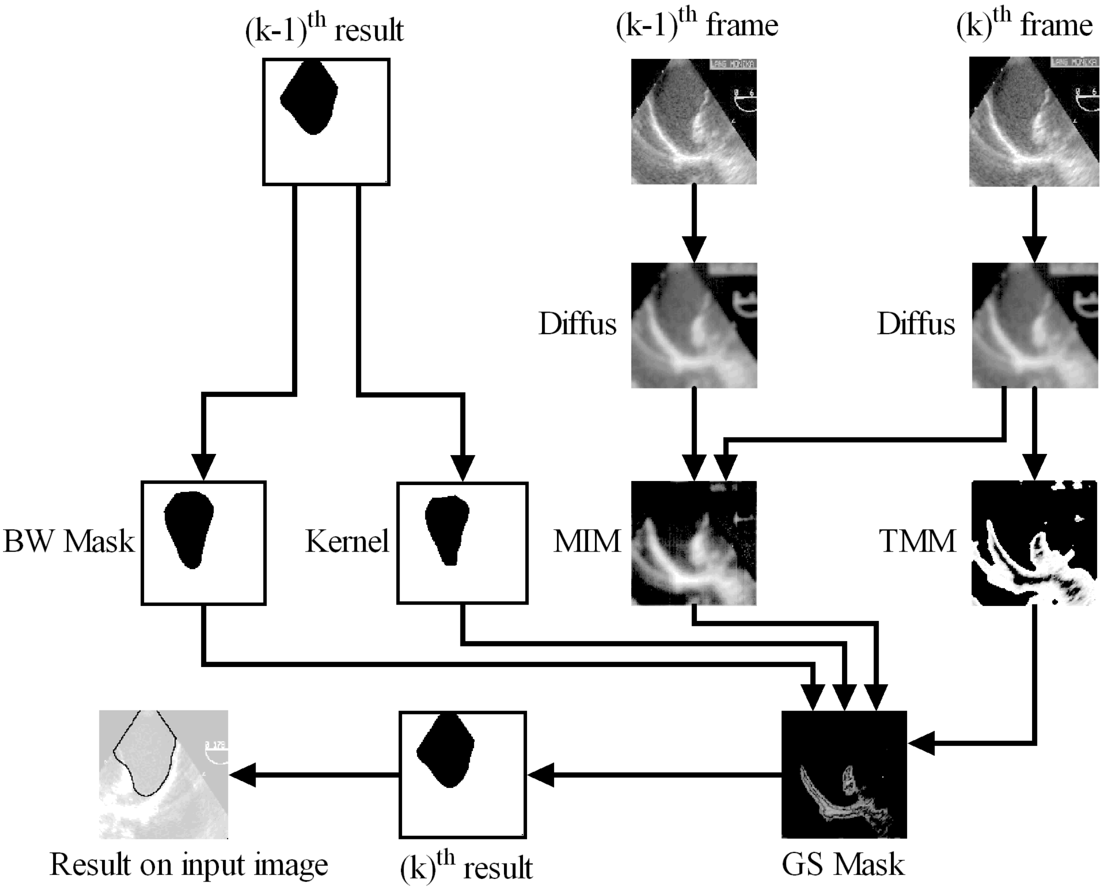


Figure 2.8: Block diagram of the CWC algorithm as implemented on the ACE16k platform.

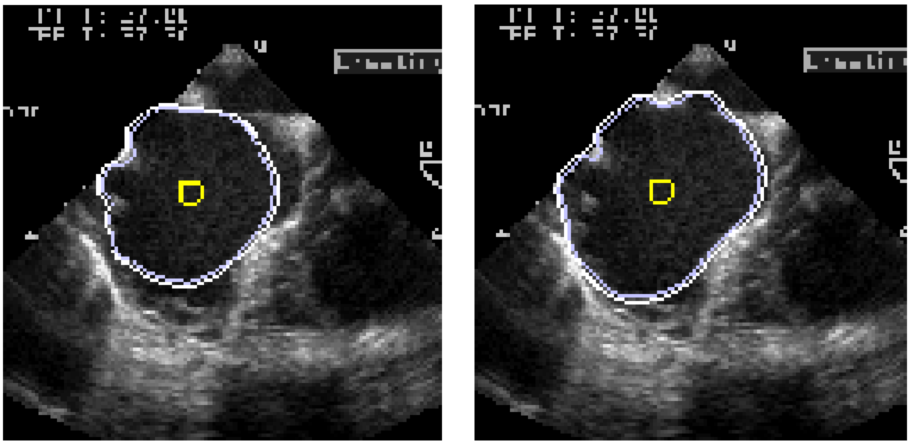
error is calculated in the Metrics module. Details concerning the functionalities implemented in the Metrics module can be found in [64].

The algorithm was implemented on the ACE16k platform [20]. Fig. 2.8 shows the block diagram of this implementation.

2.4.4 Initialization of the CWC algorithm

The behavior of Algorithm 1 when processing the first frame of a video flow is discussed (initialization or re-initialization). Assume that the system-level calibration module cuts and scales properly the window containing the region of interest. The grayscale external force map is computed exactly as in the recursive processing state. However - in the absence of a result from the previous frame - the wave source or kernel is a small, synthesized black rectangle in the center of the ellipse estimated by the sensor-level calibration module. Without a previous detection, the BW mask limiting the wave propagation cannot be calculated either. The region that falls outside the span of the ultrasound beam is known a priori. Thus this region is used as the BW mask image for the first frame (see square-pattern covered region on Fig. 2.7).

As originally presented in [40] [39], the dynamic wave propagates in an isotropic manner from the kernel. This method gives inaccurate results in the quite frequent case when the region to be detected is not approximately circular or diamond shaped. In the case of echocardiography, we can assume that the shape of the chamber to be tracked can be approximated by an ellipse (this is a quite common approximation in the medical literature for the left ventricle). To make the wave reach the cardiac walls at roughly the same time in each direction, the propagation speed must be proportional to the ratio of the major and the minor axis of the estimated ellipse. This can be done by distorting the feed-back template of the dynamic wave propagation so that the wave travels proportionally faster in the direction of the major axis. The result of the distorted wave propagation is that the evolving wave front will reach the spatial constraint at about the same time in each direction. Thus we prevent the wave from flowing out in the direction of the minor axis at the time it has not yet approached the cardiac wall in the direction of the major axis. This concept is called non-isotropic dynamic wave computation and is illustrated in Fig. 2.9. The distorted "A" template still has to fulfill the conditions described in [40] in order to ensure global propagation. In addition to distorting the feed-back template we also have to set a proper time constraint to stop the propagation. The major and minor axis estimated by sensor-level calibration module provide a first estimation. Since wave propagation starting from a square kernel and using a distorted feed-back template travels in an elongated but still diamond shape, using the minor axis as time constraint ensures that the wave front reaches the spatial constraint in every direction.



0.25	0.25	0.25
0.25	3.00	0.25
0.25	0.25	0.25

0.17	0.26	0.28
0.28	3.00	0.28
0.28	0.26	0.17

Figure 2.9: Isotropic and anisotropic wave propagation and their feed-back templates. Left side: "A" template values for isotropic propagation. Left image: result of the isotropic propagation starting from the central, square kernel. Right image: result of the non-isotropic propagation starting from the central, square kernel. Right side: "A" template values distorted corresponding the estimated ellipse. The parameters of the ellipse: Major: 36.8 pixels, minor: 21.7 pixels, theta: 0.32 radians.

2.4.5 Method for determining proper time constraint for wave propagation

The lack of spatial constraint at some wall-segments on most ultrasound video-flows requires further considerations. We have to determine a proper time constraint that will stop the wave propagation where wall segments are missing.

Assume a trigger wave operator that counts the number of currently active pixels during propagation. By active pixel we mean the number of pixels that became black in the current "iteration" during wave-propagation, equivalent to the number of pixels with state variables in the transition region in the continuous time case. The number of active pixels is measured as the Hamming distance between the patches of the previous and the current time step. In the first frame we can run the wave propagation for a long time constant. This will create a signature curve as shown in Fig. 2.10. The first valley of the curve signals that the wave arrives to the walls. This is where we have to stop the wave propagation to avoid outflows. If the spatial constraints were complete, the curve would fall to 0 meaning that no active pixels are present despite any further propagation steps. In the example shown in Fig. 2.10, the suggested time constraint with added time for filling concavities is 22 that falls very close to the minor axis of the ellipse estimated by the sensor-level calibration module (21.7).

To estimate t_d, we have to zoom at the first peak in the signature. This was done by re-launching the wave propagation from a deflated version of the previous result. As shown in Fig. 2.11, the exact time constraint can be fine tuned depending on how many time steps we make the wave propagate after reaching the maximum number of active pixels. This can be defined as the percentage of the kernel area (e.g. 5%). Like other parameters it can also be tuned to cardiologist traced contours by comparing algorithm calculated results with cardiologist traced contours.

The method described can only be solved in an iterative way on current hardware platforms. Later in the dissertation I shall suggest architectural changes that could enable this method to be implemented on future CNN architectures.

2.4.6 Method to counteract the effect of an inexactly chosen time constraint

Working with presently available DSP - CNN architectures we can measure the parameters of an ellipse using the sensor-level calibration module. Using the minor axis as time constraint can be fine in those cases where the target region does not deviate too much from the ellipse-like shape. This is however not true in many cases. Almost all cardiac ultrasound video-flows contain wall segments where the spatial constraint disappears for a few frames and then reappears again. At such places the dynamic wave will flow out of the target region. I developed a way to counteract this artefact.

From the previous detection, the contour can expand in two steps. First expand the eroded version of the detected cavity shape from the previous frame with the time constraint t_c, chosen with the algorithmic steps presented above.

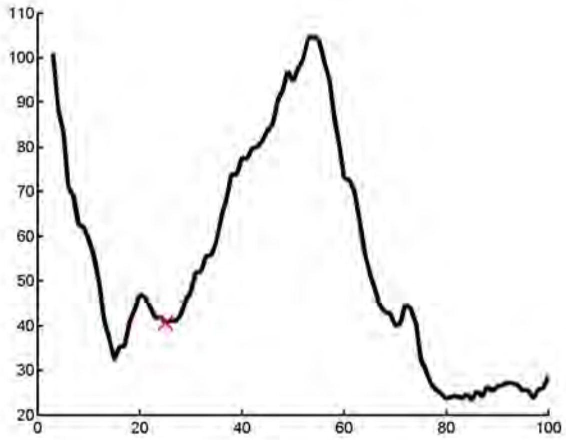


Figure 2.10: Signature created from the number of active pixels during wave-propagation. The position of the cross shows the suggested time constraint for the first frame with added time ensuring the contour arrives into eventual concavities of the cavity. Horizontal axis: time steps, Vertical axis: number of active pixels.

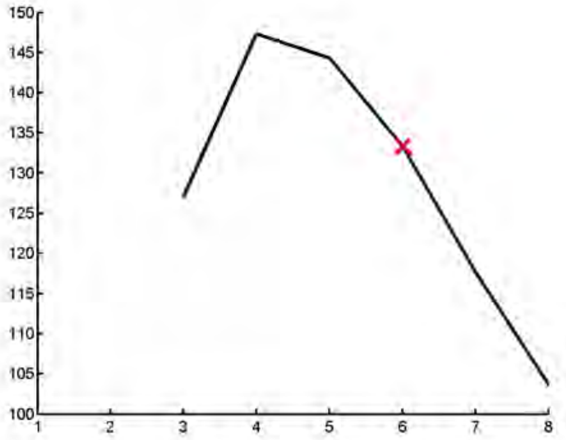


Figure 2.11: Signature created from the number of active pixels during wave-propagation. The position of the cross and the circle show suggested time constraints during the recursive state. Horizontal axis: time steps, Vertical axis: number of active pixels.

Then the expanded contour is further propagating with the same time constraint again. The second expansion results in heavy outflows where wall segments are missing. The difference between the results from the first expansion and from the second is a contour with outward protrusions where wall segments are missing. Applying the skeletonization and the pruning operators on this object corrects outflows and smoothes the result.

This method does not suppress outflows but helps the algorithm to recover when spatial constraint is reappearing. This is an important improvement since in the case of a specific cardiac disease (ASD), there is a hole on the cardiac wall where the contour must really flow out. Also, since the calibrated value of wall displacement (the value of t_b) can change during the recursive processing state, this method can also refine the result of the wave propagation to fit the spatial constraint better.

2.4.7 Pixel Level Snakes

In contrast to CWC, the Pixel Level Snakes (see the original work on PLS in [65]) method solves the contour tracking task in an iterative fashion using topographic cellular operations. The contour evolution is based on binary and local morphological operations which perform a directional contour expansion (DCE) of the active contour followed by a directional contour thinning (DCT) along the four cardinal directions. These operations are driven by guiding information extracted from the image being processed as external forces and from the contour itself as internal and balloon forces.

The external potential is derived from the input image features to establish a map of external forces. The contour will evolve in those directions where the potential field decreases. This external potential should be defined in such a way that its valleys coincide with the boundaries of the region of interest. This step is strongly dependent on the particular application PLS is used to solve and therefore represents an external input to the algorithm.

The internal potential is derived directly from the active contour itself. It represents a curvature dependent control over the evolution of the active contour to ensure its appropriate, application dependent smoothness level. The internal

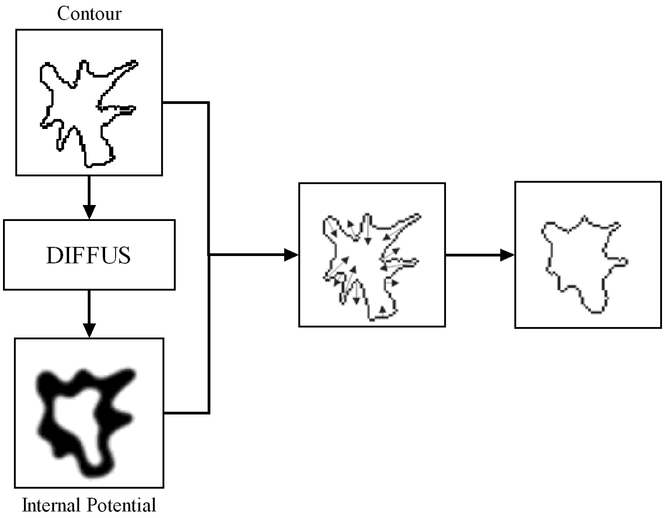


Figure 2.12: Internal Potential estimation module. It performs a diffusion operation that will originate positive internal forces that reduces local curvature

potential estimation module reaches this aim by performing recursive low-pass filtering or diffusion operation on the contour image.

In the GFE module, a directional gradient operation on the resulting image will then originate positive internal forces that will reduce local curvature or - in other words - smooth the contour shape. The effect of IPE is illustrated on Fig. 2.12.

Due to the inherent nature of curvature driven internal forces the contour has a shrinking tendency. To counteract this effect along with the necessity to trespass spurious isolated weak image edges, an additional inflating force field called balloon force is introduced. The pixel-level snakes can effectively inflate or deflate the contours by the definition of a new potential field and the consequent balloon forces. Balloon forces are calculated using a hole-filling operation on the contour image. The pixel-level snakes can effectively inflate (deflate) the contours by adding higher (lower) potential terms to those locations inside the closed curves with respect to those situated outside. See Fig. 2.13 for an illustrative definition of these notions.

In the end, by summing the weighted external, internal and inflating potentials we obtain the global potential field on which a directional gradient operation

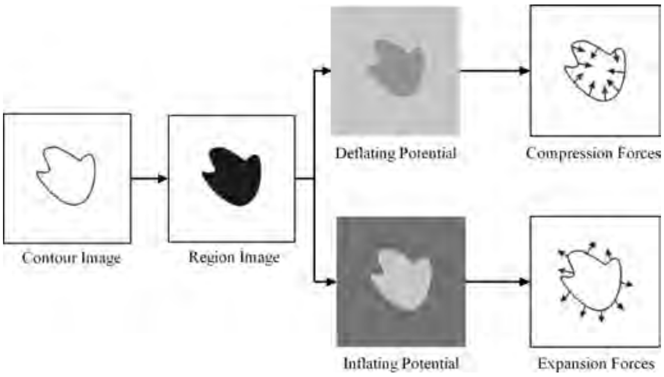


Figure 2.13: Balloon forces are calculated using a hole-filling operation on the contour image. The pixel-level snakes can effectively inflate (deflate) the contours by adding higher (lower) potential terms to those locations inside the closed curves with respect to those situated outside.

is performed providing input to the GFE module. In a pixel-level iterative technique, only the sign of the guiding forces along the direction under exploration is actually needed. The guiding force extraction module (GFE) creates a binary map with activated pixels in those locations where the potential is decreasing along the direction under study. Thus the contour evolution is allowed where this map contains activated pixels. Fig. 2.14 illustrates the operations in the GFE module.

The DCE module expands the contour in the four cardinal directions where the guiding forces are positive. The expanded contour image is processed in the DCT module to deactivate pixels that belong to locally decreasing potential and do not entail a rupture of the contour connectivity. In other words DCT makes the contour one pixel wide and well-defined.

The full featured PLS method is able to avoid uncontrolled collisions between contours. It also handles contour splits and merges during evolution. Although this can be a powerful feature in some applications, in the case of cardiographic image processing this module is not needed and is not covered here, for a detailed description of this feature see [66].

The PLS method detecting contours in cardiac ultrasound images as follows. When processing frame k, an external energy field ($I_{Ext}(k)$) is combined from two differently processed versions of the raw image (see the pseudo-code description in

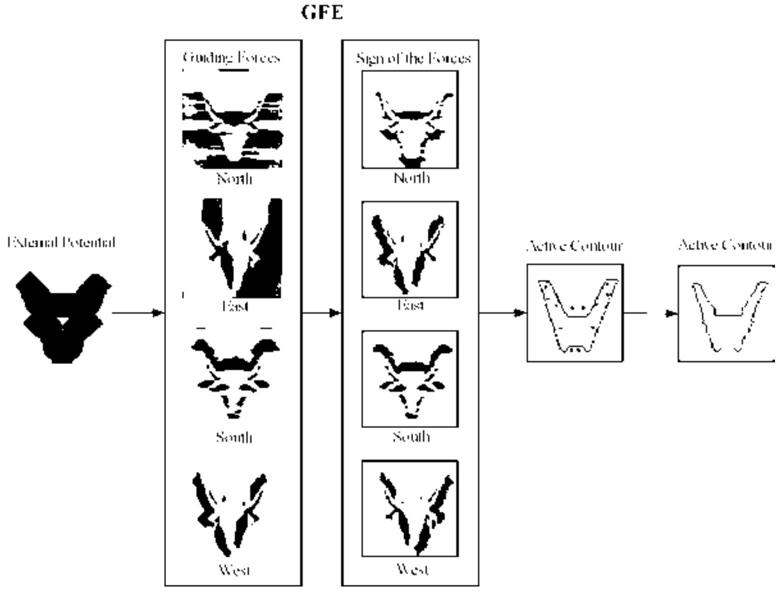


Figure 2.14: Guiding force extraction (GFE) from an external potential field. Lower potential is represented by lower intensity. By means of directional gradients the component of the guiding forces for each direction is obtained. The sign of these forces will indicate the correct direction to move the active contour.

Algorithm 2 Pseudocode of the PLS method

```
function $I_{PLS}(I_C(k-1), I_{Control}(k), w)$
    $I_{Ext}(k) = ExtEnergyEst(I_{Control}(k))$
    for each cardinal direction do
        $I_{Int1}(k) = IntEnergyEst(I_C(k-1))$
        $I_{Int2}(k) = BalloonForceEst(I_C(k-1))$
        $I_{EnergyMap}(k) = DirGradient(Combine(I_{Ext}(k), I_{Int1}(k), I_{Int2}(k), w))$
        $I_C(k) = DirContourExp(I_C(k-1), I_{EnergyMap}(k))$
        $I_C(k) = DirContourThin(I_C(k))$
    end for
end function
```

Algorithm 2). Internal energy field is obtained by applying a DirGradient operation (approximation of the Sobel operator, see Appendix for details) on $I_C(k-1)$ which will originate positive internal forces, reducing local curvature to smooth the contour shape (IEE module). To trespass spurious isolated weak image edges and to counteract the shrinking tendency of the contour, an additional inflating force field is estimated by applying a DirGradient operation on $I_C(k-1)$. The resulting force field will put the active contour closer to the boundaries of interest from locations where the external forces are too weak. Finally, the GFE module will combine the $I_{Ext}(k)$, the $I_{Int1}(k)$ and the $I_{Int2}(k)$ using user chosen weights. Then a directional gradient operation is applied to the resulting image. The sign of the gradient will indicate the direction to which the active contour has to move. GFF will be a black and white map where black pixels indicate a decreasing gradient towards the valleys of the guiding forces where the image feature to be detected is supposed to be. The DCE module is an iterative shift operation of the contour pixels detected in frame $k-1$. The shift is performed into the four cardinal directions driven by the guiding information, i.e. by the black pixels in GFF. DCE is followed by a directional contour thinning operation (DCT) deactivating the pixels in locations which do not entail a rupture of the contour connectivity.

2.5 Validation

2.5.1 Validation methods and Comparison Metrics

There is no crystallized validation protocol to assess the performance of medical image analysis algorithms. Chalana and Kim in [67] introduced a protocol for validation procedures relying on multiple observers, but debated issues persist in defining proper methods for validation [68].

Other studies compared only either the area (e.g. [37]) enclosed by algorithm and by expert traced contours or the averaged Euclidean distance between closest contour points ([36]) for validation. Another approach is to compute the distances between corresponding landmark and reference points [30]. In the present study, morphological precision was of primary importance in order to assess how our algorithm can capture the complex geometries that arise especially when the left and right atrium of a diseased heart is analyzed. Therefore as novel approach, the nonlinear wave type metric [64] was applied to measure the difference between manually and automatically extracted endocardial boundaries.

2.5.2 Metric for comparing contours

Meaningful evaluation of image segmentation requires a properly defined metric and some reference data sets. The choice of a robust metric is an intricate task. The most obvious criterion of the degree of coincidence of point sets is a measure of symmetrical difference, i.e. the area difference. This is a natural choice and it is the well-known Hamming distance which is the result of a pixel-wise XOR operation on two given finite binary point set A and B:

$$d_{Hm} = \sum (A \cup B \backslash A \cap B) \tag{2.6}$$

Another often-used distance is the Hausdorff distance. The Hausdorff distance is defined as

$$d_{Hs} = \max(h(A,B), h(B,A)) \tag{2.7}$$

where $h(A,B) = \max\limits_{a \in A} \min\limits_{b \in B} \|a - b\|$ and $\|.\|$ is some norm on the points of A and B. The function $h(A,B)$ identifies the point $a \in A$ that is farthest from any point of B and measures the distance from a to its nearest neighbor in B using the given norm $\|.\|$.

Although the Hamming and Hausdorff distances are commonly used in image processing applications for object comparison and classification, they have several disadvantages. Hamming distance measures the area difference, but does not reveal anything about shape difference. In addition, it is sensitive to object shift and noise. Hausdorff metric measures shape difference but cannot tell anything about shape properties, like average distance between two objects, e.g. a one pixel sized noisy spot can drastically modify the Hausdorff distance. In [64] the nonlinear wave metric was introduced that measures both area and shape differences between two binary objects. Let a binary wave - very similar to the constrained wave operator used in CWC - be started from $A \cap B$ and propagating till it matches the points of $A \cup B$. The time required for the wave to occupy $A \cup B$

measures the difference between the shapes A and B. During wave evolution a grayscale image d_{lHS} is created in which a pixel value corresponds to the time required for the wave to reach that pixel from $A \cap B$. The sum of these local Hausdorff distances gives the wave-type metric $d_W = \sum d_{lHS}$. In addition to capture both area and shape differences, this metric has parallel implementation with about 10 μs running time on [19] or [20].

2.5.3 Manually traced reference data sets

Gold standard for comparison is not available thus manual segmentations done by two independent experts were used. Human experts manually traced endocardial boundaries of 6 3D data sets. For all frames in a volume a mean patch was derived from the two reference contours I_{C1} and I_{C2} manually traced for the same US frame by two independent experts. Filling the contours into patches and taking the pixel-wise XOR relation of the two reference patches I_{P1} and I_{P2} only those parts that are not belonging to their intersection are kept. The result is a closed shape with thickened regions where the original patches had shape differences. A thinning operator shrunk these thickened regions into a one pixel wide mean contour I_M:

$$I_M = Shrink\left[(I_{P1}\ XOR\ I_{P2})\ OR\ I_{C1}\right] \tag{2.8}$$

Manually traced contours by the two experts were compared to I_M using the nonlinear wave-type metric to assess inter-observer variability:

$$\epsilon_{io} = \frac{1}{2 \cdot F} \sum_{r=1}^{2} \sum_{f=1}^{F} d_W(C_r(f), C_P S(f)) \tag{2.9}$$

where $f = [1, ..., F], f \in \mathbb{N}$ and F is the number of frames in the dataset. In the same way, the error of CWC ϵ_{CWC} was computed for each frame in a volume, i.e. the mean reference contour was compared to the endocardial contour extracted by the CWC algorithm.

2.6 3D reconstruction

The result endocardial boundary coordinate set detected by the TCAC algorithm is transformed into the space of the raw 3D data set. A mesh is reconstructed from the point set in 3D via the metaballs method that is an implicit surface modeling technique popular in modeling organic objects like the surface of the endocardium. Contour coordinates properly translated and rotated into their position in 3D space act as centers of force fields. This set of points are the centers of the metaballs. Around each metaball a force field is defined that is decreasing proportionally with the distance from the metaball. If the force field of all metaballs is summed, then the object surface F can be defined as the isosurface of all spatial locations belonging to the same value (T) in the summed force field:

$$F(x, y, z) = -T + \sum_i b_i \cdot e^{-a_i \cdot r_i} \tag{2.10}$$

where r_i is the Euclidean distance from a reference point in space to the center of the i^{th} metaball, a_i and b_i are parameters that define the range of influence of individual points.

The number of triangles in the smooth 3D mesh of the endocardial surface generated by metaballs is far too high for real-time interactive manipulation purposes. Despite the performance of modern graphics hardware, real-time visualization of complex meshes cannot be realized without special techniques to reduce the number of triangles that are rendered. A Mesh Reduction Method (MRM) was used to simplify the complex meshes by seeking a trade-off between preservation of original topology, high simplification speed and low number final triangles.

Endocardial boundary extraction and 3D surface reconstruction was performed for each cavity (right atrium, and part of left atrium) separately. The two atria are merged into compound mesh and any points inside the compound are deleted. The points connecting the right and left atria in the compound define the ASD surface. The area of this surface is measured and presented to the clinician on the interactive surgery planning interface.

2.7 Experiments and Results

2.7.1 Hardware-software elements

The system-level overview of the proposed method is shown on Fig. 2.15. Echocardiographic data was acquired using a Philips Sonos 5500 equipped with Omni-

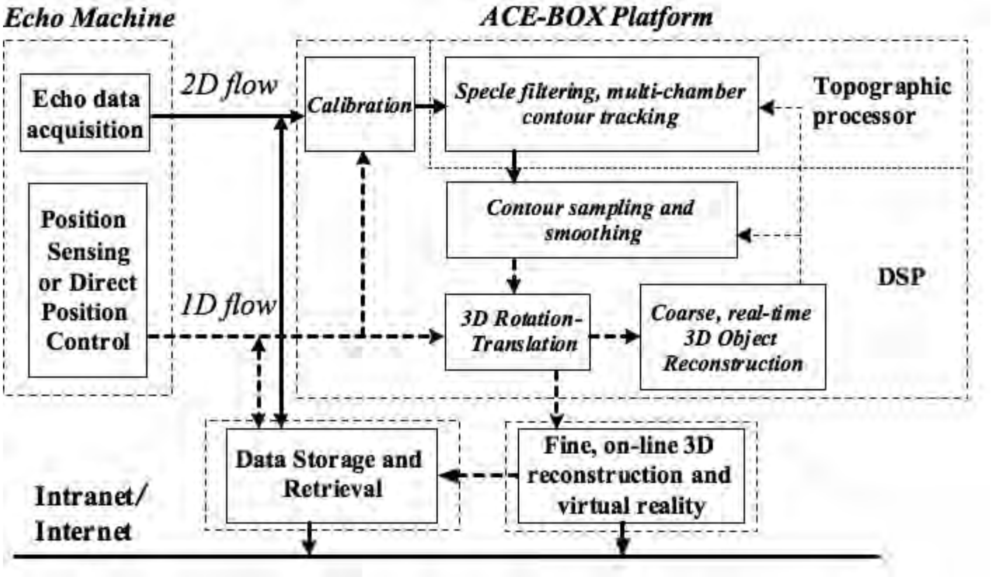

Echo Machine
ACE-BOX Platform
Echo data acquisition
2D flow
Calibration
Specle filtering, multi-chamber contour tracking
Topographic processor
Position Sensing or Direct Position Control
Contour sampling and smoothing
DSP
1D flow
3D Rotation-Translation
Coarse, real-time 3D Object Reconstruction
Data Storage and Retrieval
Fine, on-line 3D reconstruction and virtual reality
Intranet/ Internet

Figure 2.15: System-level overview showing the data-flow processing diagram of the on-line 3D echocardiographic system. US data acquisition is done by the clinician. Real-time multi-chamber contour tracking is performed on the topographic multi-processor unit, contour sampling and 3D operations are done on the non-topographic processor. The clinician can watch a coarse 3D reconstruction of the object in 3D real-time speed. Fine details can be inspected for diagnosis and surgical planning on-line. ACE-BOX stands for Analogic Computing Engine.

plane II transesophageal probe for 3D acquisition. Acquired volumes were processed on a stand-alone visual processor (Bi-i, Analogic Ltd., Budapest, Hungary) connected to a PC that provided the user interface and ran the 3D mesh generation algorithm.

To track the endocardium of a cavity, the user has to define the regions of the cavities to be processed by drawing rectangles around each cavity. The selected subregions are extracted from the 720×540 resolution raw data and the boundary of the enclosed cavities are tracked by the TCAC method in a time multiplexed way at 128×128 resolution each. A series of segmented endocardial boundaries of the RA and the LV is shown on Fig. 2.16.

After a full volume has been acquired, a coarse 3D reconstruction of the processed cavity is immediately presented to the user to give a general feedback about the quality of the acquisition. The fine 3D mesh of each cavity is rendered

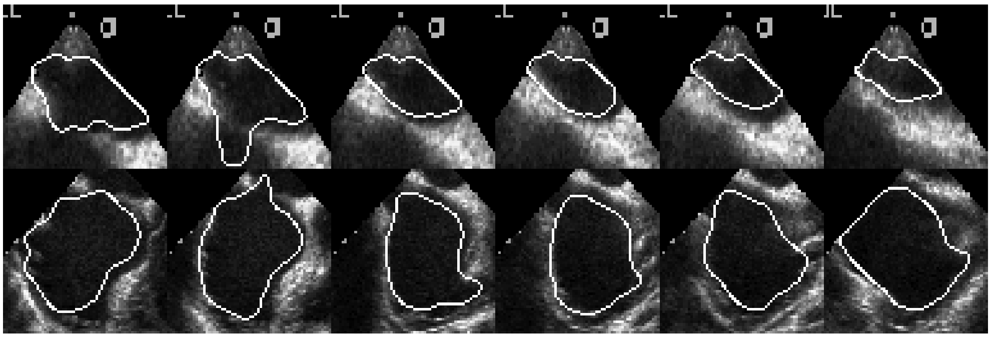


Figure 2.16: Representative snapshots of the boundary tracking of two atria, tracked by the discrete time version of CWC. Other methods provide similar results. Upper row: Boundary of the left atrium. Lower row: boundary of the right atrium. Observe in the first column how the contours flow into the other atrium through the pathological hole. The hole is most apparent in the second column and disappears as the probe rotates further in space.

to produce the virtual reality scene and accurate volume estimation in one minute after acquisition.

The continuous time version of CWC was implemented on the 128 × 128 pixels resolution, massively parallel processor (ACE16k [20]). The discrete time version of CWC was implemented in C language and processed 64 × 64 pixel sized subframes on a DSP (Texas 6202 at 250 MHz) in the Bi-i stand-alone visual computing environment. The same code was also executed on a PC (Pentium 4 3GHz with 1Gb RAM). The steps from converting the boundary coordinates into an interactive scene and calculating quantitative measurements were programmed in a commercial 3D modeling package running on a PC (3D Studio Max, Autodesk Inc., San Rafael, CA).

2.7.2 Data

6 3D data sets of in-vitro static phantoms and 3D transesophageal data sets from 6 children, mildly anesthesized and being affected by ASD were acquired with one volume per patient each containing 90 frames.

Two human experts trained to evaluate echocardiographic recordings but unfamiliar with the result of boundary tracking algorithms have traced for all frames the endocardial boundaries of the 3D data sets. To assist this work I developed a custom software tool based on the initial version developed by György Cserey. An expert had to manually delineate the first frame with as many points as he judged necessary. Since consecutive frames have a roughly similar shape, the cardiologist had the option to adapt the contour points from the previous frame to fit the current. To assess inter-observer variability, the 3D data sets were traced by two independent experts.

2.7.3 Validation of volume quantification

Volume and morphology of the RA in newborn individuals affected by ASD can vary in a relatively wide range. In lack of in vivo gold standard, validation of volume estimation on in vitro phantoms with known volumes can be a reliable alternative to in vivo validation. The volume estimation accuracy of the proposed method was evaluated on 6 in-vitro static balloons with various shapes and known volumes. Balloon objects were sealed to prevent water flowing out and immersed in a water solution. On each balloon, acquisition, boundary tracking, 3D reconstruction and volume estimation was performed.

2.7.4 Clinical case study

Pediatric cardiology is a field where 3D echocardiography has the potential to significantly improve the efficiency and reliability of diagnosis and thus may greatly improve the planning of surgical procedures. The clinical value of the on-line 3D method reconstructing the RA and the ASD was assessed in a prototype system installed in a clinical pediatric department. Fig. 2.17 shows a specific case we pursued. In the case of pediatric cardiology, an important number of children have an abnormality called atrial septal defect (ASD), i.e. a hole between the atria where oxygenated blood from the lungs is mixed with used blood coming from the main veins. Since a few years, a common practice is to introduce an implant (i.e. an occluder), that closes the abnormal hole permanently giving the patient a considerably better and longer life. The main problem is that 2D transesophageal echocardiography (TEE) echocardiography is unable to give appropriate geometrical information in many cases, i.e. the size, the morphology and the location of the defect can only be estimated with limited reliability. Therefore in some cases the type and size of the occluder is selected in an iterative fashion. Fig. 2.17 illustrates that after acquisition and reconstruction of the atria, the clinician can try a set of occluders modeled in 3D virtual reality into the heart of the patient.

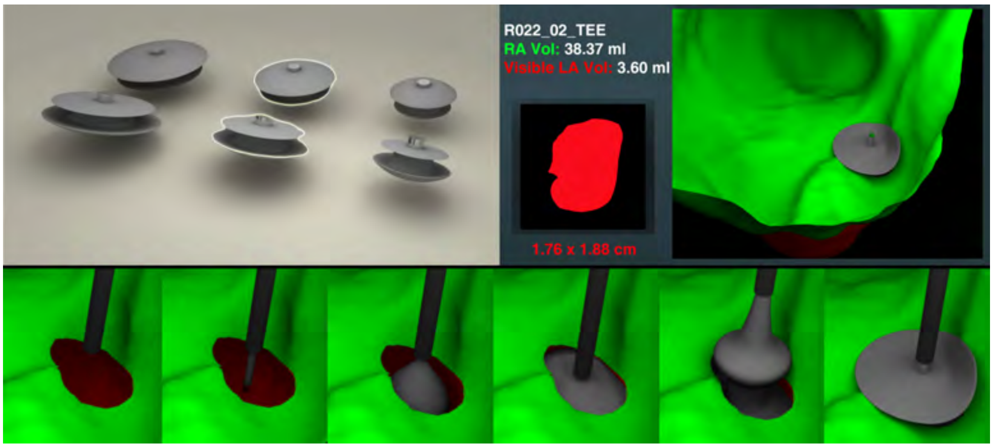


Figure 2.17: Reconstructed right and left atria in the virtual reality environment. Upper left: Occluders of commercially available sizes were modeled so that the appropriate type and size (e.g. illustrated by the yellow highlight) can be probed into the 3D model of the heart cavities. Lower row: snapshots of the process demonstrating how the selected occluder is opened in the virtual reality scene to simulate the surgical procedure. Upper right side: Quantitative measurements are shown to the user, as well as the size of the ASD. The shape of the abnormality is shown under the text in the figure. The right atrium is opened from the user's viewpoint to enable inspection of inner surfaces.

After 3D reconstruction the clinician can inspect the morphology of the selected cavities at the desired time instance and can manipulate the cavities interactively to plan the surgery (see Fig. 2.17 and Fig. 2.18).

2.7.5 Accuracy

We validated the accuracy of the on-line 3D echocardiography system using static in vitro dummy objects. To simulate a heart cavity, 6 rubber balloons with diverse geometry have been filled with water, sealed and placed in water bath. Boundary detection of each object was performed. The volume of the algorithm reconstructed object deviated from the manual reference with 12% (average across

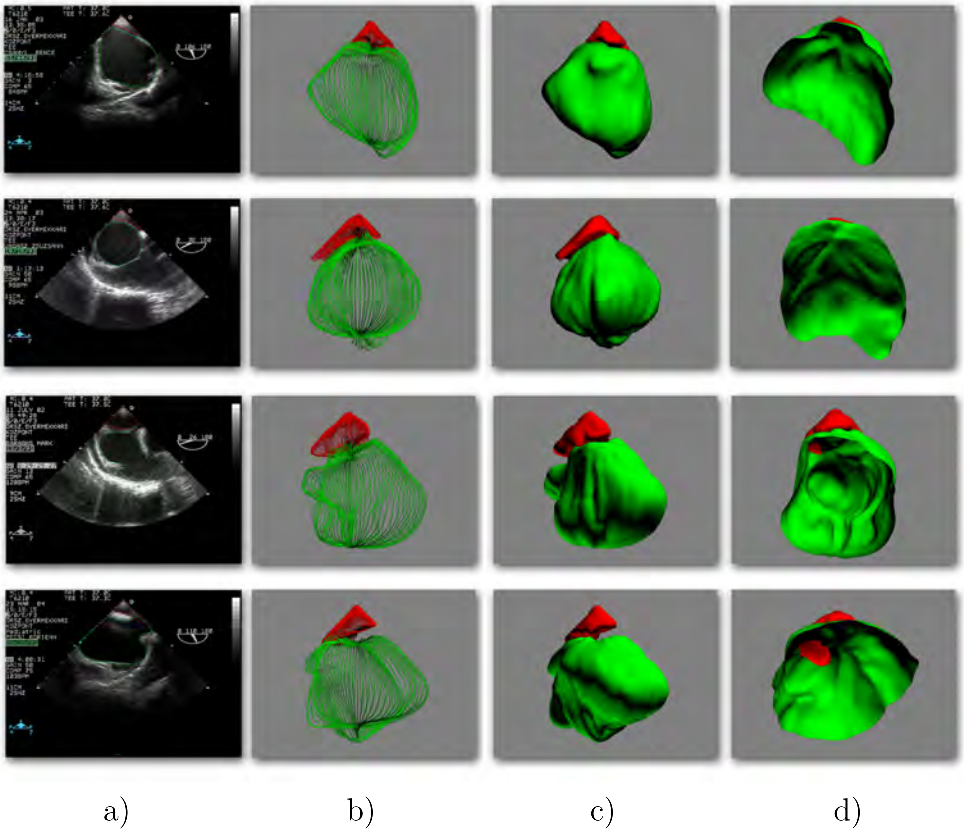


Figure 2.18: Four reconstructed clinical cases. First two rows show a healthy case, last two rows show the pathological interaction between atria. Columns: a) A sample source frame, result splines are superimposed. b) Result contours rotated and translated into 3D space. c) Result of metaball 3D surface rendering after mesh reduction d) Sample virtual reality view.

the 6 recordings). A sample dummy object, the intermediate steps of boundary tracking and 3D reconstruction can be seen on Fig. 2.19.

For 6 clinical recordings where two independent, manually traced references were available, interobserver variability of volume quantification was compared to the error between algorithm results and the mean reference. Mean error between the 6 recordings and the mean of the reference volumes was 10% whereas the mean error between individual references and their mean was 3%.

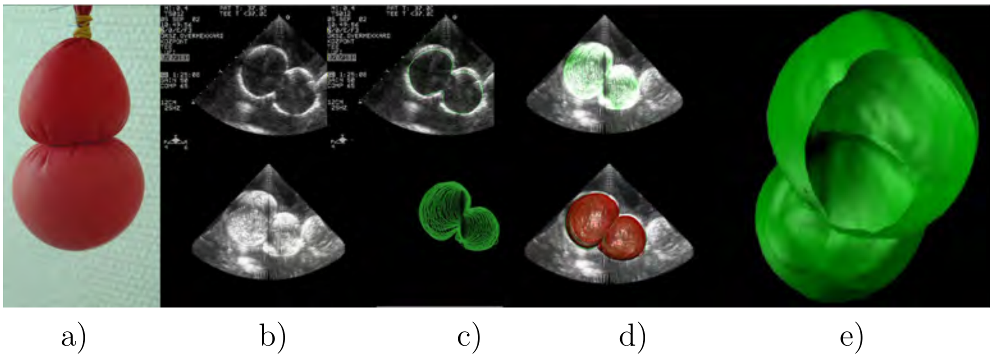


Figure 2.19: Sample case of validating the volume estimation accuracy of the system. 6 balloons with various shapes and sizes were acquired and processed. a) Original object b) above:raw US frame, below: raw 3D reconstruction c) above: sample result contour below: result contours rotated into prober position in 3D space d) above: result contours on raw 3D reconstruction, below: roughly rendered 3D reconstruction of contours on raw 3D reconstruction e) final reconstruction in the virtual reality scene.

2.7.6 Computational Performance

2.7.6.1 Comparison of topographic cellular active contour (TCAC) methods

Topographic algorithms implemented on massively parallel cellular processors outperform digital, non-topographic approaches when fast processing of image flows is needed. In the continuous time version of CWC, the so-called constrained trigger wave operator [40] is a dynamic operator that solves the contour detection problem in a single instruction. The continuous time version of CWC was implemented on the ACE16k platform and uses only 1.3 ms per frame for boundary detection.

Table 2.1 shows major theoretical aspects of the topographic active contour algorithms. Note that a priori motion information not used in PLS. This information could be important in several cases when false edges appear in the frame sequence. PLS differs from CWC representing the active contour explicitly. PLS has an inherent capability to deflate the contour or let parts of the contour move inwards. In contrast, the trigger wave operator in CWC is expansive but CWC

Property / Method	CWC continuous time	PLS	CWC discrete time
Data representation	topographic	topographic	topographic
Information exchange in space	acausal cellular nearest neighbor	acausal cellular nearest neighbor	acausal cellular nearest neighbor
Information exchange in time	read ahead recursive nearest frame	causal recursive nearest frame	causal recursive nearest frame
Contour representation	implicit (region propagation)	explicit (curve propagation)	implicit (region propagation)
Contour localization technique	expansive	expansive / contractive	expansive / contractive
Computational method	PDE related	energy related	rule based

Table 2.1: Comparison table of major theoretical aspects of our topographic active contour methods

can handle inward moving contour segments by shrinking the previous result to obtain the kernel for the current frame.

Table 2.2 presents important implementation aspects of the TCAC algorithms. When speaking about performance and quality issues, we have to keep in mind that it is extremely difficult to quantitatively compare two different active contour algorithms. A new aspect of this study is that the three types of TCAC algorithms were implemented in a common framework. Analogic architectures (i.e the ACE4k and the ACE16k chip embedded within the ACE-BOX or Bi-i environment) provide a fixed complexity reproducible CNN computing (nearest neighbor computing with linear CNN templates).

The PLS core is using linear uncoupled CNN templates and showed detection speed of 25 fps on the ACE4k.

The discrete time version of CWC was implemented using non-linear uncoupled operations running on the DSP platform of the ACE-BOX architecture, and the simplified version of the discrete time CWC was implemented on the ACE16k. Note that PLS is a general technique whereas CWC exploits the echocardiography-specific prior of approaching the boundary from the center.

Property / Method	CWC continuous time	PLS	CWC discrete time
Implementation	dynamic	iterative	iterative
Algorithm parametrization	kernel operator level (template entries)	computing block level (weights)	computing block level (threshold levels)
Minimal complexity of CNN formulation	linear coupled	linear coupled	linear uncoupled
ACE4k implementation	partial	complete	-
ACE16k implementation	complete	-	complete
ACE-BOX/Bi-i implementation	complete	complete	complete
Development motivations	echocardiography	general technique	echocardiography

Table 2.2: Comparison of implementation aspects of our active contour methods

The continuous time version of CWC is a dynamic method using linear coupled CNN templates.

This comparison shows the outstanding capabilities of using propagating templates for image processing tasks. Earlier, the extremely high performance of CWC was tempered by occasional hardware instability when running coupled templates on the ACE16k. I developed a Matlab toolbox based on the CNN template tuning method of [69]. Using the toolbox, the coupled constrained trigger wave was shown to run robustly on the ACE16k [4].

2.7.6.2 Comparison with other Approaches

Table 2.3 summarizes the speed and spatial resolution of some reviewed works together with the proposed methods. The integer performance measurements executed in a standardized test environment (SPEC®, SPECint2000, www.spec.org) were used to bring processor performances to a common ground. Content of this comparison table should be taken with great caution and it is not intended as a rigorous performance analysis. 2D and 3D methods were brought to a common

First author and reference	Keyword description of the underlying algorithm	Microprocessor	CINT 2000	Spatial resolution of input (x*y(*z))	Fps
Bosch [30]	Active appearance models	Pentium III 800 MHz	365	768*576(*16)	2,67
Mitchell [31]	3D active appearance models	Pentium III 1GHz	423	96*72	6.25
Wolf [36]	Multi-scale edge detection	Pentium III 1 GHz	423	768*576	1
Montagnat [41]	deformable surfaces	Digital PWS 500	161	256*256(*9)	0,21
Montagnat [42]	anisotropic diffusion, deformable surfaces	Pentium 4 2GHz	640	256*256(*17)	0.04†
Gérard [44]	3D deformable models	AMD Athlon 1.2 GHz	443	256*256(*40)	0,17 †
Zagrodsky [45]	3D deformable models	Dual Pentium 4 1.7 GHz	1200	128*128(*128)	0.186
Comaniciu [48]	Fusion, SA-PCA	Pentium 4 2 GHz	640	640*480	20
Proposed method	topographic, discrete time	Texas 6202 250 MHz	315	64*64(*90)	22
Proposed method	topographic, discrete time	Pentium 4 3 GHz	1149	64*64(*90)	170
Proposed method	topographic, discrete time	Pentium 4 3 GHz	1149	128*128(*90)	40
Proposed method	topographic, continuous time	Parallel (ACE16k)	Analog proc	128*128(*90)	500

Table 2.3: Computing performance comparison. Common ground for comparison is the number of processed frames per second (fps), derived from published data. Processing power of processors on which the given method was implemented were brought to a common ground using measurements from Standard Performance Evaluation Corporation (CINT2000 www.spec.org). Spatial resolution is given for reference, third number in parenthesis denotes resolution in the third spatial dimension. The reader should take this comparison with great caution and is referred to the text for further comments. Related studies where no performance figures were published were not included. †: Processor speed and type is estimated based on chips released one year before the manuscript was submitted.

ground via calculating the number of processed frames per second (fps). For 3D methods, if the published performance result was in volumes per second, the volume per second value was multiplied by the lowest spatial resolution to get fps. We are aware that even after this transformation, fps gives slightly distorted information about the performance of the original algorithm. Therefore I did not go further in transforming results to a common measure, because further spatial scale changes would not have taken into account the unknown ”slow down” ratio specific to each different algorithmic approach when processing data sets with different resolutions.

Also, the reader should keep in mind, that TCAC methods were applied to extract the right atrium whereas other methods were applied to the much simpler problem of extracting the left atrium. An early version of the CWC algorithm was already applied to segment the left atrium from 2D echocardiographic recordings, see [39] for details.

Development of an on-line 3D reconstruction algorithm that balances realistic hardware constraints with enough level of detail is an intricate task involving manual experimentations. The mesh reduction method (MRM) implemented in the Autodesk 3d Studio Max software package gave efficient support for this process, typically reducing mesh size by 1.5 order of magnitude in about a minute processing on a Pentium 4 2.4 GHz PC without noticeable topological distortions. The result is a mesh that can be explored in virtual reality on a common computer without expensive specialized hardware.

2.8 Discussion

An on-line, highly automated method reconstructing the RA and providing quantitative measurements and the size of the ASD was presented. To my best knowledge this is the first report on automated boundary extraction and 3D reconstruction of the right atrium. Quality of boundary tracking was compared to interobserver variability on 6 in-vitro static objects and 6 clinical datasets.

When comparing results of boundary detection methods (for a series of example contours see Fig. 2.16) to the interobserver variability between hand-traced contours produced by different cardiologist experts, one can hardly draw a clear conclusion. The error being noticeably higher than the interobserver variability might be explained by the ill posed nature of the problem. The atria of ASD affected patients have extremely irregular shapes that ruled out statistical shape modeling approaches. Statistical modeling are heavily used in other studies working on the LV.

Computational complexity was so far hardly mentioned in the literature as a problem. The increasing need for computer assisted, quantitative tools in medical diagnosis requires extra processing power but state-of-the-art ultrasound machines already suffer from heat issues. The advantage of using TCAC algorithms implemented on state-of-the-art topographic processors (Eye-Ris 2.0 www.anafocus.com, [62] [70]) becomes apparent when power dissipation is also taken into account.

It is extremely difficult to quantitatively compare the quality and performance of two different boundary tracking algorithms. These issues underline the need for a standardized test recording set in order to eliminate variations in algorithm performance due to variance in image quality and spatio-temporal resolution between recordings. Together with manual boundary tracings, such a set could also give solid ground to quality comparisons between various boundary detection algorithms.

Regarding TCAC methods, we went as far as possible in implementing the three boundary tracking algorithms in a common framework. Current topographic parallel cellular architectures provide a fixed complexity reproducible computing, i.e. nearest neighbor, linear interactions between cells. Performance of real-time TCAC methods ranges currently between 22 and 500 fps. At the time this dissertation was written, new generation topographic processors as well as much faster DSPs are already available indicating that the lower boundary of these figures can be improved by about one order of magnitude. This extra processing power could be used to increase spatial and temporal resolution of our approach, i.e. to move from on-line towards truly real-time operation. In addition to increased resolution, more performance could give space to quantitative measurements based on wall motion analysis. Note that the performance bottleneck is currently a major obstacle that hampers the on-line, clinically powerful implementation of other promising methods, like strain rate imaging.

In this perspective, the figures in Table 2.3 suggest that traditional computing architectures used in today's PCs have little chance to provide the computing power needed for 3D real-time echocardiography. Note that high-end echocardiography machines actually perform computations on embedded PCs. Recently the IBM-Sony-Toshiba consortium published their first-generation CELL processor, capable to run 10 threads simultaneously [71]. Although the power efficiency of the CELL processor is low, in my opinion only methods ready to be implemented on parallel hardware have the perspective to realize 3D real-time operation in echocardiography.

2.8.1 Limitations and possible extensions of the current system

The unprecedented speed of TCAC algorithms is due to the massively parallel platform they are implemented on. Availability of the chips I used is limited and their price is currently higher than that of high-end, off-the-shelf DSPs. However next generation chips, like the parallel architecture implemented on field programmable gate array (FPGA) [72], or the next versions of [73] will solve this issue.

During the experiments, I had no access to RT3D probes, therefore further efforts are needed to adapt our method to this emerging technology. Together with the MRI validation, we plan to achieve this goal in the future.

2.8.2 Novel Analogic Cellular Architecture Motivated by TCAC Techniques

There has been a number of initiatives to suggest modifications to the current ACE-BOX architecture (see e.g. [74] and references therein). I presented TCAC methods within the frame of a specific case study proposing three different algorithms running on the current ACE-BOX architecture. The aspects I add here are the condensation of this work. Three main elements - global operations, enhanced templates and more complex synaptic non-linearity - and some general system-level considerations of current CNN architecture will be discussed. My propositions aim to further enhance the power of topographic signal processing in order to improve the applicability of CNN architectures to engineering problems.

When looking at the ensemble of possible enhancements I acknowledge that factors like power dissipation and chip surface constraints do limit the freedom of hardware designers. However, exploiting the vertical dimension would be a way to alleviate the interconnect related problems inherent to high complexity CNN chips. The suggested modifications can be implemented by dividing the planar chip into separate blocks, each occupying a separate physical level interconnected by short and vertical interlayer interconnects. This would result in significant improvement in performance and reduction in wire-limited chip area, although 3D integration of transistors also faces design difficulties related to power dissipation and the associated thermal effects [75]. Unfortunately, fabrication technology of 3D integrated circuits is still quite complex and its availability is extremely restricted.

2.8.2.1 Global operations

The capability of counting the number of black pixels on the CNN chip could be a highly useful feature (see paragraphs on choosing proper time constraint for CWC). At this time this can only be done by stopping the wave propagation, sending up the full image to the DSP, and waiting for the result from the DSP. This takes far too much time. A ”global count” feature would open the space to control wave-propagation not only by the spatial constraint but also by measuring the output of the evolution. For example, a propagating template could be stopped by continuously compare its ”area” to a predefined maximum number of black pixels to be reached.

The advantage is that the global count feature does not need specific extra wiring on chip thus its hardware implementation does not eat up much of the chip surface. This feature is the same as the ”global line” idea suggested in [16], but extended with a ”continuous” read-out feature. The address event detection scheme introduced in the ACE16k processor can be considered as a first step towards the global count functionality.

2.8.2.2 Enhanced template handling support

In the case of the simplified version of the discrete time CWC or the PLS, the bottleneck is template memory. The algorithms use only a low number of template types, and some of them have 8-8 different versions corresponding to each cardinal direction. These 8-8 versions differ only in their orientation: they can be derived from a prototype (e.g. the North-facing template) by consecutive rotations. Support for rotatable and mirrorable templates could be added with only slight software modifications on the ACE-BOX driver architecture. This might also speed up the development and execution time of other iterative algorithms.

The experiments show that there is a need for a modified CNN cell structure that supports locally masked template operations. This idea can be considered as a simplified version of spatially variant templates where the source of variation comes from the grayscale image data. The transformation of grayscale levels to template levels would require level-shifting circuits with about four times the transistors per CNN cell compared to current CNN chips. In addition, for each cell some extra wiring would also be needed. In contrast, this idea is a trade-off with simpler functional capabilities but also lower hardware complexity compared to spatially variant templates.

To implement locally maskable templates on hardware, transformation of grayscale image data to local binary masks via thresholding circuits would be needed. This would only need about double as many transistors per cell compared to current implementations. Eight on-chip memories would store the result, i.e. the images determining which cells in the local neighborhood should be masked. We can think of these eight source images for local template masking as an analogy to the grayscale and binary spatial constraints in CWC guiding and limiting a global operation.

2.8.2.3 More complex synaptic nonlinearities

Median filtering, non-linear sorting and fuzzy mean level estimation are useful functionalities not supported by current hardware implementations. Advanced synapses that support radial basis function with adjustable membership function parameters would open the way for these advanced functionalities. In this study, the hardware implementation of CWC worked around fuzzy mean level estimation by iteratively thresholding the input image around the calibrated wall intensity level and summing the grayscale values under the resulted mask images. This took however more processing time and introduced instability in the system. Therefore complex non-linear synapses or even adaptive synapses implemented in hardware would highly improve the applicability of CNN architectures to a wide range of image processing tasks.

2.8.2.4 Benefits of improved data transfer rate between the CNN and the DSP platform

Experiments also suggest that combined DSP-CNN (non-topographic, topographic) algorithms could be designed that make an optimal use of the ACE-BOX type computational infrastructures. However the slow up/download transfer rate of images between the DSP and the CNN platform hampers the design of efficient combined DSP-CNN algorithms. With the use of optimal signal resolution, highest-speed state-of-the-art ADCs, the data transfer rate could be high enough to make the CNN chip be seen from the DSP platform as a memory range. Today, data transfer rate over 1 Gbit/second is quite a realistic objective. Therefore it is highly advised to care about eliminating performance bottlenecks, e.g. on the ACE16k, binary data can only be read out as a grayscale image that clearly causes a useless waste of time.

High data transfer rate would simplify algorithm implementation and would greatly improve the overall performance of the ACE-BOX architecture. In order to maximize the use of processing power in both platforms, at least a two-threaded system would be needed to enable calculations running concurrently on the CNN and the DSP platform.

Having smooth data transfer between the CNN and the DSP would also solve problems due to the limited amount of on-chip memory. In addition, this system would need less on-chip memory that would free up valuable chip surface that could be used for other valuable functionalities like enhanced synaptic complexity or template structure. Therefore in order to further improve the competitiveness of CNN platforms, it would be highly preferable to address these issues in next generation chip designs.

2.8.2.5 Remarks on some issues of the ACE-BOX architecture

Both CWC and PLS rely heavily on thresholding operations. Since the current on-chip implementation of thresholding has a not compensable noise of ±10%, it means that satisfactory results can be achieved only by using the thresholding operator implemented on the DSP that results in high transfer overhead.

I have experimented with the non-isotropic wave propagation on the ACE 16k chip. At the moment, it cannot really be used to process the first frame in CWC to ensure continuously smooth fitting to spatial constraints. The problem is that small distortions in the A template values cause no or very small amount of non-isotropy in wave propagation.

As a slightly related issue, I have tried to compensate the inevitably uneven surface properties of different chip instances. Due to hardware implementation issues, the bias map could not be used for this purpose on the ACE16k. Therefore I applied a massively distorted A template that could actually drive the wave evolution in an anisotropic way. On a more finely tunable next generation chip this idea may also be useful.

Current CNN chips suffer from the "long-term" instability of on-chip analog memory. By "long-term" we mean the time longer than one or two template operations. Although basic propagating operators can be run in a robust and reproducible way on the ACE16k platform, complex algorithms can hardly be designed due to this memory fading effect. Analog memories become noisy and unreliable after a few template operations - even if the actual template operation has not touched the memory. The introduction of special calibrating circuits [20] represents a considerable improvement in analog memory stability over the last chip versions. For those analog memories not used in the current template operation, a continuous charge-refreshing circuit could be useful to ensure their long-term stability. In summary, the exploration of topographic wave-computing on the ACEx (e.g. the slightly modified version of the ACE 16k) architectures enhanced by the above mentioned features still has great potential to improve the speed of current topographic active contour techniques.

Note that using the CNN template tuning toolbox, algorithms can be tuned to specific chip instances and [4] reported highly robust operation of complex algorithms on the ACE16k.

2.8.3 Real-time, quantitative assessment of cardiac wall dynamics

There is a growing research activity in developing sophisticated, clinically relevant quantitative echocardiographic methods. When combined with informative visualization techniques - e.g. color coding of the raw ultrasound image - these methods promise advances in helping early diagnosis of cardiovascular diseases. Methods analyzing 2D video-flows like strain rate imaging [76] have already found their way to clinical testing, although in many cases imaging artefacts still challenge clinicians in interpreting strain rate recordings [77]. In addition, analysis of strain rate recordings is still done off-line that lowers clinical throughput.

Papademetris et al. in [78] used biomechanical models to segment in-vitro dog hearts and estimate 3D regional deformation of the LV. Their model is numerically solved by finite element method that is computationally very expensive. Nevertheless, their approach and results set a desirable target for future research in quantitating cardiac wall deformations and dynamics.

Although virtual surgery is in itself an intriguing application of real-time echocardiography, we believe that real-time or near real-time techniques assessing wall motion and tissue deformations would cause a breakthrough in both the deeper understanding of heart dynamics and in early diagnosis of abnormalities. Therefore we started to analyze spatio-temporal characteristics of the cardiac wall. Results of this work are preliminary and will be presented elsewhere [5].

2.8.4 Future directions in real-time 3D echocardiography

Cardiovascular diseases account for the greatest share of mortality in developed countries. Early diagnosis, i.e. regular population screening is of primary importance in improving life expectancy. Screening is a viable approach only if the selected technique has high clinical throughput and reliable diagnostic power. Therefore the ultimate aim of research in medical image processing is to maximize clinical throughput and minimize false diagnoses. I believe that this study could represent an important step towards this aim by introducing a system that is able to monitor quantitatively the evolution of the morphology of the ASD and the volume of the RA in patients in order to assess the development of the disease.

Real-time methods are indispensable when examination time is an issue. With the advent of fast scanners the amount of data acquired is just too much to be analyzed by humans. The approach of being able to ”go back” to a patient’s data in the case when more profound inspection is needed is not available today. It takes time and the archive space is limited given the huge amount of data to be stored. Reliable model based 3D reconstruction gives both the advantage of short examination time and small data amount to be stored (a two chamber reconstruction like shown on Fig. 2.18 occupies only 0.5 Mb disk space).

Diagnostic value of a modality is a very intricate question. Today, it is a widely accepted view, that - especially in emergency situations - ultrasound is highly suitable to set up initial diagnosis giving quick qualitative information about the patient’s condition. Reliable interpretation of echocardiograms however needs highly trained cardiologist experts. In addition, for complicated situations, artifacts inherent to US modality calls for verification by more reliable but more expensive and/or invasive techniques.

Hao et al [79] segmented normal, ischemic and infarcted tissue regions from intracardiac echocardiography images using seeded region growing method. The segmentation is performed in a multifeature vector space that is constructed from texture and gray-level information. Although their approach is invasive and off-line, their study is a good example to show the power of automatic, thoroughly validated segmentation techniques.

The advantages of real-time quantitative 3D echocardiography is far from being accepted by the wide community of clinicians. In fact, most experts still exclusively use hands on analysis of 2D echocardiograms regarding the advanced 3D techniques as nice but valueless approaches. This may be due to the lack of very thorough comparisons between reconstruction methods working on cardiac MRI and on cardiac US. A very attractive work to follow was presented in [80] where ejection fraction was estimated using three modalities and their performance were compared. A similar work on 3D reconstruction methods could provide strong evidence for the clinical value of 3D real-time US reconstruction methods.

RT3D recordings have the advantage of high temporal resolution, portability and affordability. computer tomography (CT) or MRI I offer high resolution anatomical recordings. The fusion of these, i.e. acquisition of RT3D images registered to a couple of anatomical slices could reveal the dynamics of the heart at quality and speed that could reveal clinically valuable findings. The solution of this issue falls out of the scope of our current work but needs more attention and investigation.

2.9 Conclusion

My work reports for the first time a complete 3D echocardiographic diagnostic system that exploits the processing power of scalable, pixel-level parallel TCAC algorithms. The system is capable of multi-chamber reconstruction and tracking that enables to study the interaction of cardiac cavities. The ability to visualize the interaction of multiple chambers is a unique, novel aspect of this study. The clinical value of this approach was demonstrated in a unique clinical case study assessing the morphologies, volume and quantifying the interaction of the left and right atria.

Validation of our method was done on 6 in-vitro static objects and 6 clinical samples where each volume contained 90 frames. Though six recordings does not seem too much, we have to take into account that analysis of the right atrium in 3D echocardiography is quite an unexplored field. Actually, the systematic collection of these data together with other recordings not directly used in the development (all in all 60 patients with two recordings per patient) can also be considered as a very important outcome of our efforts.

An additional novel aspect of the study was to apply non-linear wave metric to evaluate the performance of boundary tracking methods that measure both area and shape difference between expert traced and algorithm computed endocardial boundaries.

Based on the results of this work I conclude that the 3D echocardiography - even when using relatively low-cost, electronically rotated 2D transducers - with high temporal resolution can be combined with specialized hardware solutions and 3D tools that have great potential in clinical practice. RT3D systems promise a significant step forward assuming that the spatial resolution of these probes will improve in the near future.

One of the main reasons why 3D echocardiography struggles in finding its way into clinical practice is the lack of an acceptable compromise between quality and processing speed. Parallel algorithmic solutions relying on multi-core hardware architectures described in this study could help improving both:

- close sensor image processing can provide real-time feedback in order to enhance the image acquisition process through local adaptation and
- hardware supported topographic flow processing methods can easily alleviate the processing bottleneck even in case of sophisticated methods.

Chapter 3

Morphology based method for the classification of ganglion cells in the mammalian retina

Neural morphology is an important predictor of function in living neural networks. Many parts of the nervous system are organized in multiple laminae, each incorporating a unique set of cell types with unique functionality. The IPL of the mammalian retina is comprised of about ten different strata [81], formed by the dendritic arborization of a dozen different functional classes of ganglion cells. So far, tedious manual analysis was needed to estimate the depth of ramification of a Retinal Ganglion Cell (RGC). Automated quantification of biological features is very difficult due to the high variance in the morphology of biological features and artefacts caused by the acquisition procedure.

I developed a novel method for automated, quantitative estimation of the depth of dendritic ramification of RGCs from confocal image stacks.

3.1 Acquisition protocol

Dual channel confocal stacks from mice retina were analyzed where three subtypes of ganglion cells were labeled with green fluorescent protein (GFP) through retrograde trans-synaptic viral labeling (GFP channel) by collaborating biologists.

The 405, 488 and 633nm laser lines of a Zeiss laser scanning microscope (LSM) 510 Meta confocal microscope were used to scan 170 GFP labeled ganglion cells. All cell nuclei were counterstained with 4',6-diamidino-2-phenylindole (DAPI) stain to label the border of the IPL (DAPI channel).

The improvement on previous scanning protocols was twofold: an automatic stage control algorithm [82] was applied that can automatically scan a set of stacks at manually marked positions. In each imaging session, the positions of $20 - 30$ ganglion cells per retina were marked manually in about 30 minutes and then all confocal stacks were acquired unattended. This improvement created a workflow with much higher throughput than manually performing each scan.

Second, based on the observations during the development of the quantification method, the scanning parameters of the protocol were modified so that the retina is scanned above the GCL and well into the photoreceptor layer. Before, many scanned cells could not be quantified automatically because the extraction of the ganglion cell layer (GCL) and/or inner nuclear layer (INL) landmark features was not feasible in a robust way.

3.2 Method

The task was to develop a fully automated algorithm that quantifies the depth of dendritic ramification for each labeled ganglion cell from the DAPI and GFP stained cells recorded by a confocal microscope. In a scan containing the target, GFP stained ganglion cell, robust quantification is achieved by extracting four types of landmark features:

- from the fluorescence signal of the GFP stained ganglion cell
  - the ganglion cell soma
  - the local dendrite feature
- from the fluorescence signal of DAPI stained cells
  - the ganglion cell layer (GCL) border
  - inner plexiform layer (INL) border

The depths of dendrites in the stack are determined relative to the GCL border and the INL border. Fig. 3.1 shows that the GCL border (0% depth) is defined as the depth of the peak DAPI fluorescence intensity in the GCL and the INL border (100% depth) is defined as the depth where the DAPI florescence was 66% of the maximum measured in the INL (a robust feature across various scan settings). Dendritic depths are calculated locally near each dendritic segment to eliminate spatial misalignment artifacts caused by the fact that the retina is not entirely flat.

The problem is not trivial as the scanned retina is not entirely flat, so simple projections would provide no information about the depth (see illustration in Fig. 3.1). Moreover the images have noise and other types of artifacts and the GCL border extracted from DAPI signal is only present in cell nuclei.

To overcome these limitations an algorithm was designed based on the following considerations:

- Dendrites are elongated thin image features. A steerable convolution filter can exploit this shape prior to enhance dendrite like features and suppress artefacts with different shape.

- In the ramification layer of the IPL, mostly the distal branches of the dendritic tree can be found. Neuronal function was found to correlate with the depth of the ramification layer(s). The proposed algorithm differentiates between proximal and distal dendrites. The idea is that proximal dendrites run rather vertically so their length in a given z section is shorter than that of the distal dendrites running mostly laterally. Thus extracting dendritic lengths in each z section and then collecting detections in a histogram can provide a solid method that discards proximal dendrites and keeps those that form the ramified end of the dendritic tree.

- The non-flat nature of the retina can be corrected. The x, y, z position of each ganglion cell nucleus in the GCL can be extracted from the DAPI channel. "Drilling" a virtual hole along the depth (z) direction in the recording around reference x, y positions extracted from GCL ganglion cell nuclei is used to find dendrites locally under each nucleus. In this way the waviness problem is solved as we detect depths locally.

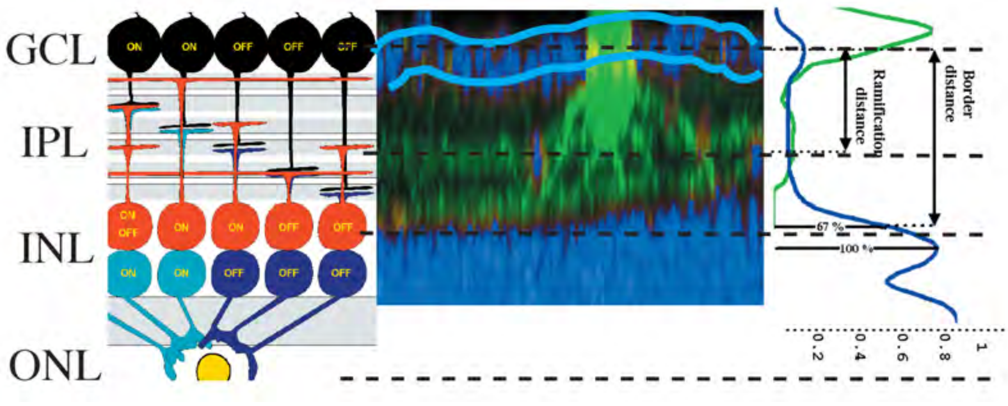


Figure 3.1: Left: A drawing of the retina. GCL: ganglion cell layer, IPL: inner plexiform layer, INL: inner nuclear layer, ONL: outer nuclear layer. Middle: Side projection of a GFP labeled ganglion cell in mouse retinal optical slice counter-stained with DAPI. Right: Fluorescent profile of the GFP and DAPI stain along the z direction. Note that due to the many artifacts (cell body, retina is not flat, etc...), one cannot see the dendritic stratification on the GFP plot. This is the reason the proposed algorithm was developed.

- Reducing the effect of artefacts at each local detection step, statistics of many local detections will provide reliable global detection, i.e. the ramification depth. Having a number of localized DAPI fluorescence intensity projections along the z direction and a number of dendrite depth values, the mean and standard deviation of the ratio of INL border distance and dendrite distances can be calculated. This ratio quantifies the ramification depth(s) of the GFP stained ganglion cell.

I conceived a 3D image processing method and implemented it to quantify the relative depth positions of the four landmark types.

High variance in the morphology of biological features and artefacts caused by the acquisition procedure make the extraction of the landmarks difficult. Robust landmark extraction method was conceived via careful analysis of depth neighborhood relations of the landmark features. The most reliable feature - the ganglion cell soma - is extracted first together with initial guesses for the GCL and INL depths.

3.3 Implementation

A detailed description of the method follows. Algorithmic solutions are summarized in pseudocodes where subroutines either have a ready implementation in most image processing software packages or are straightforward to implement. A short description of these subroutines are included in the Appendix and the interested reader is encouraged to contact the author (hillier@digitus.itk.ppke.hu) to obtain the Matlab source code of the algorithm.

The flowchart of the proposed algorithm is shown in Fig. 3.2. Detailed algorithm description of each box follows in the next section. In summary, processing of a confocal recording starts with a calibration step that gives a rough estimate of the cell soma position in the GFP channel and of the GCL and INL layer borders of the DAPI channel. The x, y coordinates of GCL cell nuclei are extracted and subregions around the nuclei (GCLwindows) will be processed further. In the subregions the waviness of the retina is negligible thus the z coordinate of each nucleus can be used as a local reference for ramification measurement. The individual GCL cell z positions (GCL) are used to restrict the z range of possible INL detections. GCL positions also serve as a reference for dendrite position detections. The final ramification depth is determined from the histogram of two distance sets: the distance of the local INL z position to the local GCL z position and the distance of the local dendrite position(s) to the local GCL position.

3.3.1 Data

A number of confocal stack data are processed by the algorithm. Each stack is composed of a number of plane images with resolution of 512×512 pixels. About 100 of such frames were acquired along the third dimension. The planar images will be referred to as frames in the x, y plane and the third dimension will be referred to as z position. The IPL has a total depth of $20 - 30$ microns.

3.3.2 Algorithm

A fully automatic algorithm is presented. The algorithm starts with giving a rough estimate to the position of the ganglion cell soma, the INL and the GCL.

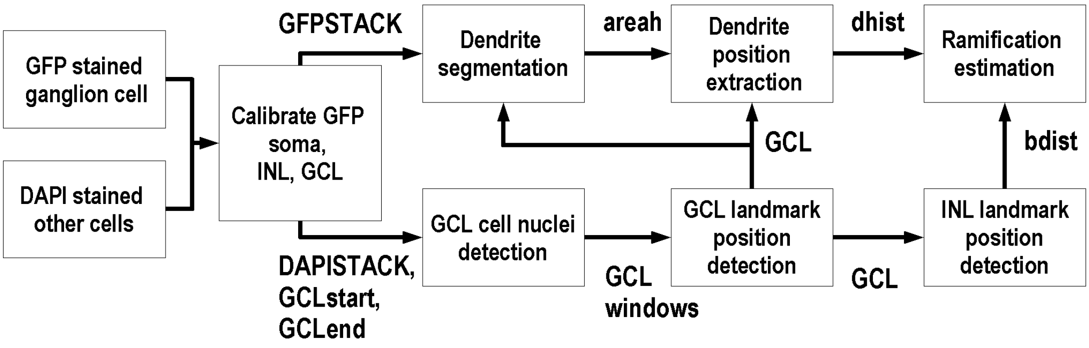

GFP stained ganglion cell
DAPI stained other cells
Calibrate GFP soma, INL, GCL
GFPSTACK
DAPISTACK, GCLstart, GCLend
Dendrite segmentation
areah
Dendrite position extraction
dhist
Ramification estimation
GCL
bdist
GCL cell nuclei detection
GCL windows
GCL landmark position detection
GCL
INL landmark position detection

Figure 3.2: Flowchart of the proposed algorithm. Each box has a detailed algorithm description in the text. The variable storing the results of a processing step is shown close to the arrows. Calibration first gives a rough estimate of the cell soma position in the GFP channel and of the GCL and INL layer borders of the DAPI channel. The x, y coordinates of GCL cell nuclei are extracted and subregions around the nuclei (GCLwindows) will be processed further. In the subregions the waviness of the retina is negligible thus the z coordinate of each nucleus can be used as a local reference for ramification measurement. The individual GCL cell z positions (GCL) are used to restrict the z range of possible INL detections. GCL positions also serve as a reference for dendrite position detections. The final ramification depth is determined from the histogram of two distance sets: the distance of the local INL z position to the local GCL z position and the distance of the local dendrite position(s) to the local GCL position.

The first step is to detect which channel contains the stained cell and which the surrounding cells as detailed in Algorithm 3.

The first channel of the stack is thresholded to keep the 1% brightest pixels. Spot noise is eliminated using morphological opening. A lower bound on the size of the soma (*minsize*) in a frame was obtained from the database of recordings. The apriori knowledge that the soma shape is close to a circle is exploited. The size of an object in a frame is calculated as the area of the object weighted by its eccentricity. This way axons and proximal dendrites having high area are not extracted with the soma. Also, objects smaller than *minsize* are not processed further.

The soma of the cell in the GFP channel spans several frames. Artefacts introduced by the LSM or by the staining process may introduce e.g. lower

Algorithm 3 Calibrate soma, GCL, INL

```
function CALIBRATION($STACK$)
    $binSTACK = Threshold(STACK, 0.01)$
    for $f = 1 : framenum$ do
        $binSTACK(f) = bwmorph(binSTACK(f), 1, 'open')$    ▷ eliminate noise pixels
        if $GlobalSum(binSTACK(f)) > minsize$ then    ▷ few pixels in a frame means no soma here
            $ms = regionprops(bwlabel(binSTACK(f)))$
            $keeplist = ms.area/ms.eccentricity > minsize$
            if $keeplist\ is\ not\ empty$ then
                $maskSTACK(f)$ = pixels of $binSTACK$ where $keeplist$ is $TRUE$
            else
                next  f
            end if
        else next  f
        end if
        for $o = 1 : objects\ in\ keeplist$ do
            if centroid3D is not empty then
                $existingobj = distance(ms.centroid(o), centroid3D) < minsize$
            else
                $centroid3D = ms.centroid(o)$
            end if
            if $existingobj\ is\ not\ empty$ then
                $centroid3D(existingobj) =$
                $= mean(ms.centroid(o), centroid3D(existingobj))$
            else
                add ms.centroid(o) to centroid3D
            end if
            keep track that this object in centroid3D was detected in frame f
        end for
    end for
    for o = 1 : objects in centroid3D do
        for $f = 1 : framenum - 2$ do
            if centroid3D(o) is in f but not in f+1 or f+2 then
                seedpoint = mean(centroid3D(o))
                limits = max(boundingbox(centroid3D(o)))
                maskSTACK(f:f+1) = imreconstruct(BinStack, seedpoint, limits)
                set f,f+1 of this object to the track record in centroid3D
            end if
        end for
    end for
end function
```

intensity frames in the middle of the soma. Applying fixed threshold and size constraints is fast but may fail to extract the soma where LSM artefacts occur in a stack. The position of the soma in the z direction is modeled as the mean of a Gaussian fitted to the sum of pixel intensities in a frame along the z direction. This projection curve was extracted by Algorithm 3.

In frames where the fixed threshold extracted a soma object smaller than $minsize$, the centroids of neighboring frames are used to complete the soma object. As the algorithm goes down frame by frame in z direction, centroid positions of newly extracted objects in the current x, y plane are compared to the x, y positions of already extracted centroid positions from the previous frames. If the current centroid is inside the area of a previously detected object, the current object is registered as the continuation of that object. Else, it is registered as a new object. Note that this method attributes objects to existing ones even if there is a gap in z direction between them. If the number of missing frames in a 3D object is smaller than 3, a seed point is placed at the centroid position and the object is reconstructed from the original stack.

Algorithm 3 extracts data corresponding to large, circular objects and increases its smoothness in z direction, i.e the projection curve in the z direction is corrected to resemble more to a Gaussian that in turn will increase the precision of subsequent detection steps. The detection of the cell soma is at the beginning of the classification algorithm, loss of precision here would propagate down the algorithm.

Algorithm 3 assumed the GFP channel is the first in the stack. If for any reason the GFP data is in the second channel, the soma detection algorithm will not find any object. In such a case the channels are swapped and soma extraction algorithm is rerun.

The z coordinate of the center of the soma is obtained by taking the global sum of each frame in $mSTACK$ along z direction. The result is a curve from which the most prominent maximum is extracted via Algorithm 4. The important parameter in Algorithm 4 is $nmax$, the number of maxima to be found. In projection curves, the underlying biological data can be noisy thus a lot more local maxima can be present than defined in the model. The number of detected maxima is reduced via smoothing the curve by the local regression using weighted

Algorithm 4 Max/min point extraction

```
function ExtractMaxMin($curve, nmax, maxiter, \alpha$)
    for $i = 1 : maxiter$ do
        $curve = smooth(curve, \alpha)$
        $[maxvals, maxinds] = peakdetect(curve)$
        if $elements\ of\ maxinds <= nmax$ then
            $[minvals, mininds] = peakdetect(-curve)$
            $return$
        end if
        $\alpha = elements\ of\ maxinds - nmax$
    end for
end function
```

linear least squares and a 2^{nd} degree polynomial model (LOESS) method [83]. The value of α defines the proportion of data used in each fit. It is called the smoothing parameter because it controls the flexibility of the LOESS regression function. Large values of α produce the smoothest functions that wiggle the least in response to fluctuations in the data. The smaller α is, the closer the regression function will conform to the data. Using too small a value of the smoothing parameter is not desirable, since the regression function will eventually start to capture the random error in the data.

An initial guess for α can be made from the z dimension of the raw stack. Depending on the noisiness of the data more than $nmax$ maxima might be found. In such cases α is automatically increased by the algorithm till the number of maxima found equals $nmax$.

The position of the soma and its size in z direction is determined by fitting a Gaussian on the filtered data of Algorithm 4. The position of the soma is a reliable marker to separate the GCL and INL borders landmark features in the DAPI channel.

The soma is assumed being a unique object in the 3D stack. In contrast, the z positions of the other landmark features are determined locally at many data sites dispersed in the 3D stack. The problem is that in the worst case, the z position of the GCL at one extremity in the x, y plane of the mounted retina can be equal to the z position of the INL at the other extremity of the retina. Therefore the

GCL and INL positions cannot be determined from just one z projection of the DAPI channel.

The solution could be to detect landmark features at local x, y regions but reliable local reference points have to be defined. Cell nuclei in the GCL could be candidate reference features and convolving the stack with a spherical kernel could extract their positions. Convolving the whole stack would be a time consuming step. Instead, a calibration step was conceived that provides the same information without 3D operators.

The whole stack is divided into four square regions in the x, y plane and the first estimation of the GCL and INL positions is calculated from projections made from these substacks.

Image features corresponding to the INL cover the whole retina thus have to be detected before GCL features that are much sparser. However, after a rough estimate of INL and GCL border positions is obtained, the detection order changes. The reason is that GCL position is based on finding the z position of cell nuclei that are sparse but robust image features. In contrast, INL position is extracted from intensities averaged frame by frame around detected cell nuclei of the GCL. Again, landmark feature detection order is determined by the quality of the image features. In the beginning, we do not know where to look for reliable local information (GCL cell nuclei), thus the globally available but less reliable INL position must be determined first. Algorithms 5 and 6 summarize the detection steps.

Algorithm 5 INL position detection

```
function INLDETECT($curve, thr$)
    $[locmaxv, locmaxi, locminv, locmini] = ExtractMaxMin(curve, 3, 10)$
    $[locmaxi, locmaxv]$ =drop locmaxi falling right to global max
    if exist(locmaxi < globmaxi) then
        calculate difference between all locmaxv and locminv
        keep those that are higher than the average difference
        [locmaxi locmaxv] = rightmost locmaxi
    end if
    $INLpos = crossing(curve, locmaxv * thr,' ascent')$
    $INLpos = max(INLpos)$
end function
```

Algorithm 6 GCL position detection

```
function GCLdetect($curve, thr$)
    $[curve, locmaxv, locmaxi, locminv, locmini] = ExtractMaxMin(curve, 3, 10)$
    $[dcurve, dlocmaxv, dlocmaxi, dlocminv, dlocmini] = ExtractMaxMin(d/dz \cdot curve, 3, 10)$
    $dlocmini = max(dlocmini < INLpos)$
    $dlocmaxi = max(dlocmaxi < dlocmini)$
    $GCLmaxi = locmaxi(locmaxi > dlocmaxiANDlocmaxi < dlocmini)$
    $GCLstarti = GCLmaxi - 2 \cdot (GCLmaxi - dlocmaxi)$
    $GCLstopi = GCLmaxi + 2 \cdot (dlocmini - GCLmaxi)$
end function
```

Algorithms 5 and 6 provide initial estimations on landmark positions. Initial estimations are used to define regions in the 3D space where precise landmark detection will be performed. Subregion based processing has two advantages: faster processing and lower number of artefacts occurring in the extraction process. In the end, the dendrite ramification can only be quantified at those regions of the x, y plane, where all the landmarks are present at their own z position.

First, the GCL landmark features are located locally. GCL cell nuclei have the most clearly detectable x, y, z positions. Algorithm 7 shows the overview of how the x, y positions of cell nuclei are detected. The last line indicates that a square window is defined around each nucleus with an a priori defined size. This size was defined so that regions remain non-overlapping but large enough to provide enough data sites with all landmark features present. The DAPI channel will be further processed only along the subregions defined by the windows.

Algorithm 7 GCL cell nuclei detection

```
function GCLCellDetect($DAPISTACK, GCLstart, GCLend, wwidth$)
    GCLplane = normalize(sum(DAPISTACK(GCLstart:GCLend)))
    GCLplane = convolve(GCLplane, 2DGaussiankernel)
    GCLcenters = regionalmaxima(GCLplane)
    GCLwindows = fitwin(GCLcenters, wwidth)
end function
```

The GCL border and INL border landmark features are extracted from the projection curves made along the z dimension in each subregion as illustrated in

Fig. 3.3. This step is detailed in Algorithm 8 and 10.

Algorithm 8 GCL landmark position detection

```
function GCLPosDetect($DAPISTACK, GCLstart, GCLend, GCLwindows$)
    GCLCURVES = sum in each frame(DAPISTACK(GCLstart:GCLend,:,:), GCLwindows)
    for $i = 1$ :number of curves $\in$ GCLCURVES do
        $[curve, maxv, maxi, minv, mini] = ExtractMaxMin(GCLCURVES(i), 1, 10)$
        calculate difference between all maxv and minv
        keep those that are higher than the average difference
        maxi = leftmost maxi
        $GCLpos(i) = GCLstart + maxi$
        $GCLend(i) = min(mini(find(mini > maxi)))$
    end for
    $[gcgx] = hist(GCLpos, max(GCLpos) - min(GCLpos))$
    $[maxvmaxiccminvmini] = ExtractMaxMin(gc, 3, 3)$
    $[GCL] = DropOutliers(GCLpos, maxv, maxi, cc, minv, mini)$
    [gce gx] = hist(GCLend,max(GCLend)-min(GCLend))
    [maxv maxi cc minv mini] = ExtractMaxMin(gce,3,15);
    [GCLEND] = DropOutliers(GCLend, maxv, maxi, cc, minv, mini)
end function
```

Algorithm 9 assumes a histogram can be modeled as a main Gaussian separated by local minima from noise contributions. In the case of GCL detection, the true GCL cell nuclei form the main Gaussian and the much less densely detected cells in the IPL form the smaller Gaussian.

Why is such a step needed? We never cannot be sure from the absolute z position of a cell nucleus that it belongs to the GCL. Subsequent INL and dendrite position detections also contain variabilities and artefacts that introduces noise in their detected z positions. The quantification of ramification uses local reference points, the GCL cell nuclei. If artefacts of reference point extractions are not handled, the error will smear the distribution of subsequent INL and dendrite detections.

The border of the GCL has just been extracted for each subregion, now the locally less reliable INL border landmark positions can be extracted in each subwindow. Algorithm 10 details the process, rather similar to the GCL border landmark detection. The intensity curve of the INL border in each subregion is

Algorithm 9 Clean point set from outliers

```
function DropOutliers($dhist, maxv, maxi, chist, minv, mini$)
    calculate difference between all maxv and minv
    [maxv maxi minv mini] = keep prominent maxv minv (higher than mean difference)
    lmini = max(mini < maxi)
    rmini = min(maxi < mini)
    dhist(dhist < lmini OR dist > rmini) = 0
end function
```

Algorithm 10 INL landmark position detection

```
function INLPosDetect($DAPISTACK, GCLEND, GCLwindows$)
    INLCURVES = sum in each frame(DAPISTACK(GCLEND:frnum,:,:), GCLwindows)
    for $i = 1$ :number of curves ∈ INLCURVES do
        [inl globmax] = INLDetect(INLCURVES(i), 0.66)
        INLPOS(i) = GCLEND(i) + inl
        PHOT(i) = GCLEND(i) + globmax
        bdist(i) = inl - GCLEND(i)
    end for
    [bc bx] = hist(bdist,max(bdist)-min(bdist))
    [maxv maxi cc minv mini] = ExtractMaxMin(bc,2);
    calculate difference between all maxv and minv
    [maxv maxi minv mini] = keep prominent maxv minv (higher than mean difference)
    INLc = min(maxi)
    [pc px] = hist(PHOT,max(PHOT)-min(PHOT))
    [maxv maxi cc minv mini] = ExtractMaxMin(pc,2);
    calculate difference between all maxv and minv
    [maxv maxi minv mini] = keep prominent maxv minv (higher than mean difference)
    PHOTc = mean(maxi)
    bip = mean(INLc, PHOTc)
    INLPOS = INLPOS < bip
end function
```

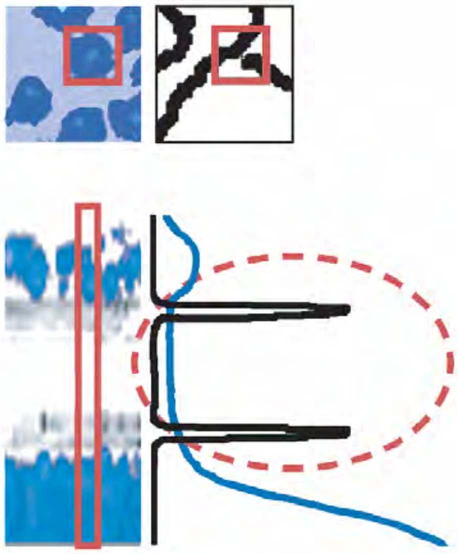


Figure 3.3: Upper Left: Detection of a ganglion cell nucleus in the DAPI stained sections. Note that the algorithm automatically finds the correct sections in the stack that contain the ganglion cell nuclei. Upper Right: a processed GFP section under the nucleus. Lower Left: we ”drill” a virtual hole in the retina (red box) under each nucleus and plot the fluorescence for both channels as a function of depth (Lower Right). We calculate the depth of dendritic ramifications from these curves by detecting the peaks in the black (dendritic) curve and the borders in the blue curve.

modeled as the mixture of a Gaussian - the INL - and a plateau like region - the photoreceptor layer. The difference of subregion based processing compared to the earlier rough detection is that here we cannot be sure that the INL peak is clearly detectable in all subregions. Therefore, the photoreceptor layer is also detected. The mean position of the INL is determined by calculating the histogram of relative INL positions. This is either a single Gaussian or a two Gaussian mixture like histogram, and in both cases the Gaussian with the lowest z position is the INL. The photoreceptor layer appears as a plateau, thus cannot be detected with similar precision. Lower precision is tolerable since the aim is to exclude false INL detections. INL detections are kept that have lower z position than the mean of the INL-Photoreceptor layer border.

Having a good estimate of the GCL border and the INL border, the range of frames along the z direction is limited for dendrite segmentation and z position detection.

Thinking of each frame as a gray-level landscape, dendrite structures appear as ridges. Following the ideas and using the implementation of [84], dendrites are enhanced in each frame as follows. Ridge-like image structures are well detected by means of second-order differential operators. Specifically, the local principal ridge directions at any point in an image are given by the eigenvectors of the second-derivative matrix computed from the intensity values around that point. Because of the symmetry of this matrix, the eigenvectors are orthogonal, with the eigenvector corresponding to the smaller absolute eigenvalue pointing in the longitudinal direction of the ridge. Comparing the eigenvalue magnitudes, the GaborFilter step computes for each pixel in a frame a measure of "dendriteness". The output of this detector, when applied to the image on the left in Fig. 3.4, is shown on the image on the right in Fig. 3.4.

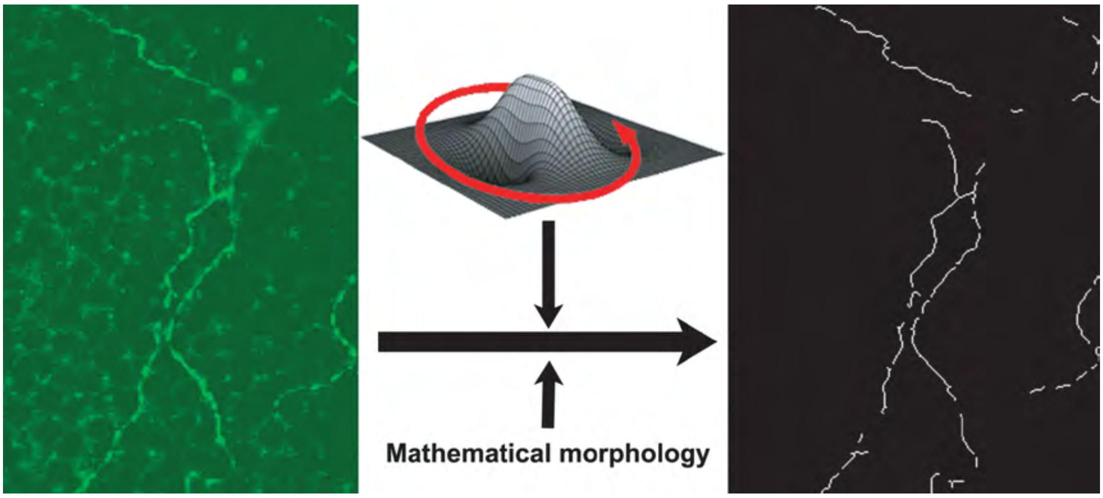


Figure 3.4: Left: an original unprocessed image from the stack. Note the low signal to noise ratio. Right: the detected dendrite after Gabor filtering and applying nonlinear mathematical morphology operators. Importantly unlike in most imaging software the detection of the dendrites is fully automatic requiring no parameter adjustments or user input. This makes it possible to process hundreds of stacks automatically.

The result of the steerable convolution filter is a normalized image with maximum intensity values attributed to regions with perfect match to a filter direction. The result image can be thresholded with a rough, fixed threshold that separates clearly dendrite-like structures from smeared dendrites or other features. After every frame were filtered and thresholded, the whole 3D subregion is skeletonized. Skeletonization also emphasizes that the exact threshold value does not affect the precision of z position detection.

Objects that have a 3D pixel count lower than a threshold ($areathr$) are discarded. The x, y plane - each frame - is subdivided into a number of overlapping square subregions. The number of subregions was obtained via experimentation taking into account the following aspects. First, when the subregion size is larger than the length of distal dendrites in a frame then no direct relation between subregion size and dendrite length in a subregion can be defined. Such a relation is required to define the size threshold $areathr$ used to differentiate between distal and proximal dendrites or artefacts. Second, when the subregion size is too small, dendrites and artefacts cannot be separated using the skeleton pixel count measure. Third, subregion based processing is required to prevent the smearing effect caused by the waviness of the retina.

In each frame of the remaining stack, the pixel count of each object is stored. This process builds a histogram of dendrite lengths along the z direction. The segmentation and dendrite histogram calculation is shown in Algorithm 11.

Ramification is quantified using the z position of a GCL cell nucleus as reference. Thus the extraction of dendrite z position is performed in each of the $GCLwindows$ as detailed in Algorithm 12. The size of subregions around GCL cell nuclei are smaller than the size of subregions defined for dendrite segmentation. This algorithm corrects the waviness of the retina at each GCLwindow via aligning dendrite length histogram to each individual GCL cell nucleus' z position. Where GCLwindows intersect more than one dendrite region, alignment is performed for each affected dendrite region.

As a final step, the local measurements are statistically evaluated to estimate the relative depth of dendrite arborization, see Fig. 3.5 and Algorithm 13. Here, we do not know the number of strata where the dendritic arbor ramifies. The model is a mixture of an unknown number of Gaussians. Maxima and minima

Algorithm 11 Dendrite segmentation

```
function DendriteSegment($GFPSTACK, GCL, regionlims$)
    for $r = 1 : numberofregions$ do
        for $f = mean(GCL) : max(INLPOS)$ do
            frame = GaborFilter(GFPSTACK(f,regionlims(r)))
            bGFPSTACK(f,regionlims(r)) = Threshold(frame, 200)
        end for
        sGFPSTACK = bwmorph(bGFPSTACK,'skeleton')
        ms = regionprops(bwlabel(sGFPSTACK))
        dGFPSTACK = sGFPSTACK(ms.Area > areathr)
        for $f = mean(GCL) : max(INLPOS)$ do
            areah(f,r) = sum(dGFPSTACK(f,regionlims(r)))
        end for
    end for
end function
```

are extracted from the dendrite histogram. If the number of maxima does not correspond to the assumed number of ramification strata, the estimation is redone with an increased number of strata in the model.ation, see Fig. 3.5 and Algorithm 13. Here, we do not know the number of strata where the dendritic arbor ramifies. The model is a mixture of an unknown number of Gaussians. Maxima and minima are extracted from the dendrite histogram. If the number of maxima does not correspond to the assumed number of ramification strata, the estimation is redone with an increased number of strata in the model.

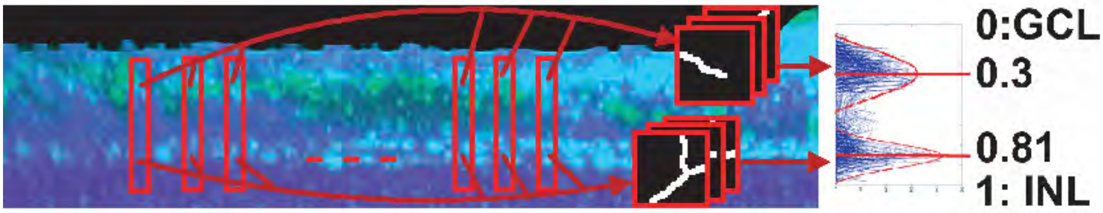


Figure 3.5: Left: Obtaining the average depth of dendritic ramification from individually "drilled" holes in the retina (marked by vertical red rectangles) under each nucleus in the GCL. The depth of dendritic ramification is calculated in each subregion by detecting peaks in the intensity profile of the fluorescent stain. Right: the distribution of normalized dendritic depth curves.

Algorithm 12 Dendrite position extraction

```
function DendriteExtract($areah, GCL, GCLwindows, regionlims$)
    pad = max(GCL) - min(GCL)
    for $g = 1 : numberofGCLwindows$ do
        for $r = 1 : numberofregionlims$ do
            if GCLwindows(g) is fully in regionlims(r) then
                dhist(g) = LocalizeDendriteHist(areah(:,r), GCL)
            else if GCLwindows(g) is partly in regionlims(r) then
                for n = regionlims intersected by GCLwindows(g) do
                    temphist(n) = LocalizeDendriteHist(areah(:,n), GCL)
                end for
                dhist(g) = mean(temphist)
            end if
        end for
    end for
end function
function LocalizeDendriteHist($areah, GCL$)
    offs = GCL(g) - mean(GCL)
    if offs < 0 then
        dhist = cat(zeros(-offs), areah, zeros(pad + offs))
    else if offs > 0 then
        dhist = cat(areah(offs:end), zeros(pad + offs))
    else
        dhist = cat(areah, zeros(pad))
    end if
end function
```

Algorithm 13 Ramification estimation

```
function RamificationEst($dhist, bdist$)
    dhist = sum(dhist)
    for i = 1 : maxstrat do
        [maxvi maxii cci minvi Rminii] = ExtractMinMax(dhist,i,5);
        if elements of fmaxii <> i then
            strati = -1*ones(i)
        else
            strati = maxii / mean(bdist)
        end if
    end for
end function
```

3.4 Results and Discussion

The 405, 488, and 633 nm laser lines of a Zeiss LSM 510 Meta confocal microscope were used to excite DAPI, Alexa 488, and Alexa 633 stains respectively. To determine the dendritic depths of ganglion cells, confocal stacks of 170 ganglion cells were acquired with an automatic stage controlled by Auto Time Series Macro software [82]. In each imaging session, $20 - 30$ ganglion cells per retina were marked manually, and confocal stacks were acquired at each location. A $63x$ 1.4 numerical-aperture oil-immersion lens (Zeiss) was used. The z steps were $0.2 - 0.35$ mm. The scan started at the ganglion-cell layer and continued until the photoreceptor layer. Landmark features for dendritic depth quantification were extracted locally to eliminate artifacts caused by the fact that the retina is not entirely flat.

All of 170 analyzed GFP-expressing ganglion cells in the left retina were found to have dendrites in only two IPL strata at depths of 30% (±4%) and 89% (±6%).

Could this result be achieved with other methods? A few other studies (see e.g. [85], [84], [86], [87]) already approached the problem of automated segmentation and analysis of dendrite (vascular) structures. Many of them are available in free or commercial image analysis packages. Similar to the validation problem in echocardiography, these approaches have not been validated on a public database built using a well reproducible data acquisition protocol.

As this study shows, the quality of the underlying data is an obvious limit of the applicability of a method. Samples from our dataset were analyzed with an automated 3D neuron tracing method [85]. It seems, that automated tracing methods are designed to process stacks acquired with well established protocol and high contrast. The tracing method could only find proximal dendrites indicating that our problem was more difficult than the capabilities of the tracing method. This issue was also addressed in [84] where the aim of fully automated segmentation was found too ambitious. There, initial dendrite detections were connected manually assisted by a shortest path finding algorithm.

The contributions of my method are as follows. The problem was difficult since

- the acquisition protocol was not well established

- in the recordings, dendrites have low contrast including gaps
- the retina is not flat.

My solution provides an automatic method that quantifies dendritic ramification of retinal ganglion cells at high throughput via

- exploiting local shape and intensity priors
- defining the detection order of landmark features
  - data sites were located based on global information
  - high contrast detections limited the detection regions of lower contrast landmark features
- outlier rejection in each landmark detection step
  - kept the precision of region limits high
  - increased the precision of final ramification depth quantification.

3.5 Conclusion

A fully automatic algorithm has been developed that can detect the depth of dendritic ramifications for retinal ganglion cells from confocal stacks. This algorithm can be used [2] as a tool to quantitatively analyze the morphology of genetically labeled neural subtypes in the retina and correlate structure with function.

Chapter 4

Synchronization in oscillator arrays

4.1 Introduction

Synchronization of oscillator networks is a prevalent phenomenon in nature. Despite its widespread presence, synchronization is used only in a few specific fields of engineering, e.g. communication with chaotic lasers.

In arrays of coupled dynamical systems, full (or total) synchronization usually denotes the phenomenon when all individual systems synchronize. Cluster synchronization means that the network breaks down into several coexisting groups of cells and each cell in a group synchronizes with each other cell in the group but not with any other cell in the array. The term partial synchronization is sometimes used as a synonym to cluster synchronization but more often the term partial synchronization refers to the situation when not all state variables synchronize or there is some kind of coherence between cells but their state vectors are not identical. We shall use the term single cluster synchronization to denote the case when only one group of cells synchronize in the network and the rest exhibits desynchronized behavior.

Understanding the principles of synchronization phenomena occurring in networks of coupled chaotic oscillators is very difficult due to the non-linear, high-dimensional nature of these systems. Studies mainly focusing on deriving conditions for synchronization include [88], where global symmetries of a network of identical, diffusively coupled oscillators were used to classify linear invariant manifolds corresponding to cluster synchronization regimes. In [89] and [90] the link between graph topology and stability of global synchronization was clarified. In [22] it was shown that stability of cluster synchronization regimes depends on the individual oscillator dynamics and on the choice of state variables used in the coupling. However the existence of synchronized clusters depends only on the coupling topology, boundary conditions and the number of oscillators.

Another issue is the dependence of the stability of synchronization manifolds on the mismatch of individual oscillator parameters. Cluster synchronization regimes were reported to persist with 10-15% mismatch between parameters of individual oscillators [22]. A common aspect of the above mentioned studies is that the coupling was the same for all oscillators.

For networks with asymmetric connectivity - i.e. where coupling varies from cell to cell - methods based on calculating the eigenvalues of the connectivity matrix [91] for determining coupling values needed for synchronization may be difficult to apply. In [89] a graph theory based method was introduced to estimate the value of the coupling coefficient needed for global synchronization of a network. This was further elaborated for asymmetrically coupled networks in [92] with the restriction that every node has the same input and output weight sum. In [93] a method for constructing a coupling scheme for arbitrarily selected n-cluster synchronization was presented for networks with non-nearest neighbor connections.

The above mentioned studies focused on providing a means to estimate coupling coefficients of the network in order to ensure complete synchronization of the cells. However, synchronization can be exhibited in a variety of additional forms including phase-[94], lag-[95] and generalized [96] synchronization. In addition, perfect and imperfect synchronization can also be differentiated within these forms of synchronization. This high complexity motivated the development of new methods that can measure synchronization in a wide variety of forms. One such measure of synchronization was proposed in [97], where a normalized coherence function based on the cross Fourier spectrum of two chaotic oscillators were calculated from time series. This metric has direct physical meaning and highly mature spectrum evaluation methods exist for its efficient and robust calculation. However, extending it to describe synchronization phenomena between more than two oscillators may be a complex task.

In [98] it was shown that two Lur'e systems which may even have different qualitative behaviour (e.g. limit cycle versus chaos or stable points versus chaos) can be synchronized to each other up to a small synchronization error. Chua's circuits, multi-scroll circuits and networks consisting of such cells with linear coupling belong to this class of Lur'e systems.

I reported a new kind of synchronization in oscillator networks with local, diffusive coupling where the coupling can be different for each cell. The reason to allow any interaction pattern - restricted to contain non-negative weight values - is that new kinds of cooperative behavior may be possible that were not observed in previous studies when coupling was kept identical for all cells.

This chapter is organized as follows. In Section 2 the oscillator network model is specified. In Section 3, methods used to find cluster synchronization regimes are presented. In Section 4 a new phenomenon is presented where highly asymmetric interaction weights can give rise to multi- or single cluster synchronization regimes with partial synchronization. Conclusions are given in Section 5.

4.2 Array of oscillatory cells

Consider an array composed of autonomous nonlinear oscillators:

$$\dot{\mathbf{x}}_{r,c} = \mathbf{F}(\mathbf{x}_{r,c}; \Theta_{r,c}) + \sum_{k,l \in S_\rho(r,c)} A_{r,c} P(\mathbf{x}_{k,l} - \mathbf{x}_{r,c}), \;\; r, c = 1, ..., M \in \mathbb{N}_0 \qquad (4.1)$$

where $\mathbf{x}_{r,c} \in \mathbb{R}^d$ represents the state vector of the r, c^{th} oscillator; $\mathbf{F} : \mathbb{R}^d \times \mathbb{R}^q \to \mathbb{R}^d$ is a nonlinear function depending on parameters $\Theta_{r,c} \in \mathbb{R}^q$ that define the cells; $A \in \mathbb{R}^{M \times M}$ is the spatially varying coupling weight matrix that defines the interaction pattern of the array; $P = (p_{i,j}) \in \mathbb{R}^{d \times d}$ with $p_{i,j} \in \{0, 1\}$ determines which state variables couple the oscillators and $S_\rho(r, c)$ defines a fixed neighborhood topology with neighborhood radius ρ.

In the experiments shown, individual cells were modified Chua's oscillators generating n-scroll attractors. N-scroll generating chaotic oscillators were introduced in [99] generalizing the original Chua's circuit by introducing additional breakpoints in the nonlinear resistor. In the current experiments hyperbolic tangent non-linearity was used [100], with state equations :

$$\frac{d}{dt}\begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -\Theta_{r,c} & -\Theta_{r,c} \end{pmatrix}\begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -\Theta_{r,c} f(x^1) \end{pmatrix} \tag{4.2}$$

where $f(x^1) = \sum_{j=-V}^{W} (-1)^{j-1} \tanh q(x^1 - 2j)$, $V = 1, W = 3, q = 2$. The qualitative behavior of an oscillator cell is show on Fig. 4.1.

The phenomena of multi- and single cluster synchronization are studied on square arrays with nearest neighbor connectivity ($\rho = 1$) and zero-flux boundary conditions. Time evolution of each array configuration was simulated in Matlab with time horizon large enough to analyze its long-term stationary behavior ($T = 9000$ with relative error tolerance 10^{-6}). Bifurcation parameter $\Theta_{r,c}$ may be different for each cell in an array. Variable $x^1_{k,l}$ was coupled to $x^3_{r,c}$, i.e. $P = [0\,0\,0;\ 0\,0\,0;\ 1\,0\,0]$ in Eq. (4.1).

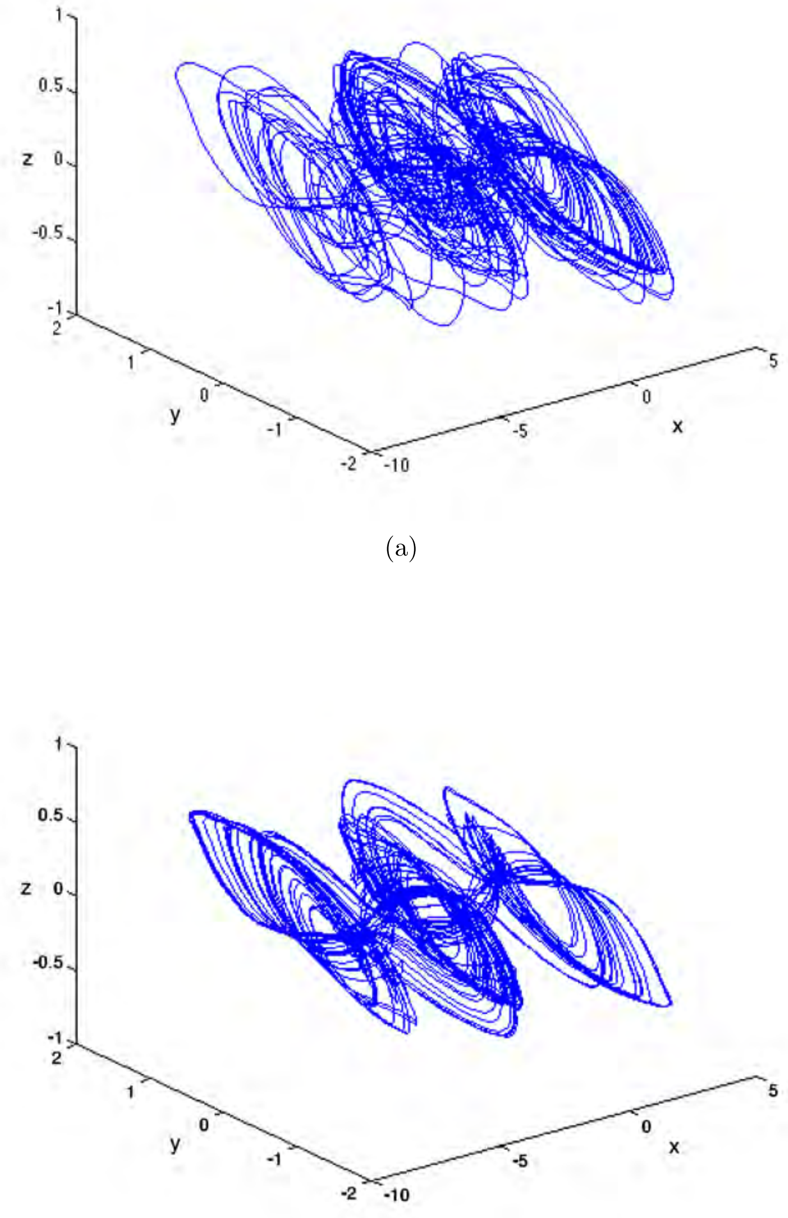
4.3 Global Optimization Framework

Inspired by [101] and [98] the problem of finding cluster synchronization regimes was cast into an optimization problem:

$$\min_{A,\Theta} U(\{\mathbf{X}_{stac}(t_0), \mathbf{X}_{stac}(t_1), ..., \mathbf{X}_{stac}(t_n)\}) \tag{4.3}$$

where $U(\cdot)$ denotes the cost function and $\mathbf{X}_{stac} = (\mathbf{x}^1(t), \mathbf{x}^2(t), ..., \mathbf{x}^d(t)) : \mathbb{R} \to \mathbb{R}^{M^2 d}$ is a stationary solution of Eq. (4.1) for a given initial condition and $t_0 < t_1 < ... < t_n < \infty$. Additional inequality constraints may apply to A and Θ. $U(\cdot)$ is constructed in such a way that dynamical properties of individual cells or cell populations fulfill the desired requirements. These requirements may include different types of synchronization and stability criteria, desired Lyapunov spectrum or embedding dimension of the attractor, etc. We assume that $\mathbf{F}_{r,c}$ can provide a rich enough set of dynamical behaviors.

An important advantage of the proposed approach compared to other studies is that the only requirement on the vector field defining individual oscillators is that they permit the solution to exist and be unique. In this study we used



(a)

(b)

Figure 4.1: Figure 4.1(a) shows the attractor of cell $(2, 1)$ corresponding to the configuration of Fig. 4.5(a). Figure 4.1(b) shows the attractor of the same cell when coupling is removed from the network. All cells in all experiments exhibit similar qualitative behavior.

oscillators with continuous vectors fields. On the other hand, in some cases this liberty may result in an optimization problem that is very hard to solve if it is possible at all. The global optimization framework used to learn network configurations corresponding to cluster synchronization regimes is generic, i.e. no assumption was made about network size or topology.

The choice of optimization algorithm is an intricate task. In this study the time evolution of the network was used to drive the learning process. Integrating the network for a given parameter set lasts long, thus a key issue is to choose an optimization algorithm that requires a low number of function evaluations to find the global optimum corresponding to the desired behavior. In [102] a new global optimization algorithm called Coupled Simulated Annealing (CSA) was introduced. In CSA the annealing temperatures of several Simulated Annealing processes are interconnected in order to improve performance in convergence speed and to increase the probability of exploring all basins of attraction in a given number of cost function evaluations. The number of cost function evaluations per individual optimizers decreases exponentially when the number of optimizers is increased linearly. This makes CSA a good candidate for the current problem since interactions between solutions decrease the number of cost function evaluations to reach a given energy threshold.

In order to learn cluster synchronization regimes, a task of primary importance is to define a proper cost function which assigns the desired behavior of the network to the global optimum. The system-level diagram of the framework is shown on Figure 4.2.

4.3.1 Synchronization Metric

The cost function has to embed a metric that

- is able to capture different forms of synchronization (complete-, phase-, lag-, generalized),
- provides a measure in a way that does not need any human interpretation, i.e. can be embedded in the cost function,
- is computationally feasible.

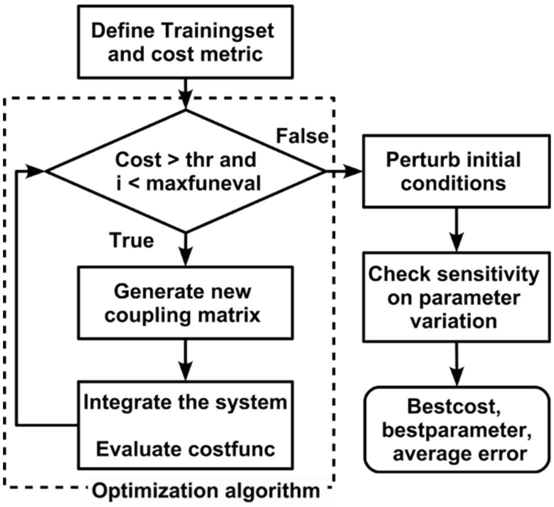

Define Trainingset and cost metric
Cost > thr and i < maxfuneval
False
True
Perturb initial conditions
Generate new coupling matrix
Check sensitivity on parameter variation
Integrate the system
Evaluate costfunc
Bestcost, bestparameter, average error
Optimization algorithm

Figure 4.2: Overview of the investigated global optimization framework that teaches a network of chaotic oscillators to match qualitative behavior requirements formulated in the cost function.

The following metric that fulfills these requirements was proposed and used in our study:

$$U_{std} = \sum_{t_j=1}^{n} \sum_{i=1}^{d} \log \frac{\left(\frac{1}{N_S} \sum\limits_{k \in \Pi_S} \left(x_k^i(t_j) - \overline{x}_S^i(t_j) \right)^2 \right)^{\frac{1}{2}}}{\left(\frac{1}{N_R} \sum\limits_{k \in \Pi_R} \left(x_k^i(t_j) - \overline{x}_R^i(t_j) \right)^2 \right)^{\frac{1}{2}}} \tag{4.4}$$

where $x_S^i(t_j) \in \Pi_S$ represents a number of snapshots taken from the time evolution of cells to be synchronized, $x_R^i(t_j) \in \Pi_R$ represents those that are not to be synchronized. $\Pi_S, \Pi_R \subset \{1, ..., M\} \times \{1, ..., M\}$ denote cell populations with N_S standing for the total number of cells to be synchronized and N_R for the number of cells to be desynchronized and $\Pi_S \bigcap \Pi_R = \emptyset$. $\overline{x}_S(t_j)$ is the spatial average for each time instance of the target population of cells to be synchronized and $\overline{x}_S(t_j)$ is the average for the rest of the cells.

The numerator of Eq. (4.4) is the distance of state variables of all cells to be synchronized with respect to the mean of their individual state variables, evaluated in corresponding time instances. Minimizing Eq. (4.4) leads to complete synchronization of cells to be synchronized. However, to achieve partial synchronization, the rest of the cells must be as desynchronized as possible from each other and from the synchronizing cells. The denominator stands for the same kind of distance as the numerator, but it is calculated for the cells to be desynchronized. The role of the denominator is to make state variables of desynchronizing cells as uncorrelated as possible with any other cell.

Given a specific cluster synchronization regime to be learnt, Eq. (4.4) assigns the full and all cluster synchronization regimes to local optima. The cost value corresponding to these local optima decreases as the position of synchronizing and desynchronizing cells better correspond with the imposed cluster synchronization regime. On Fig. 4.3 and Fig. 4.5, cells imposed to synchronize in the cost function are marked with dashed rectangles. However, synchronized behavior also appears within other cell clusters for the same array configuration.

4.3.2 Cost function imposing qualitative behavior

We consider chaotic, limit cycle and equilibrium point qualitative behavior. Lyapunov exponents are widely used to measure qualitative properties of a dynamical system. The cost function used to impose any of these modes on either the population required to be in sync or on the rest is:

$$\begin{aligned} U_{lyap} = \sum_{j\in\Omega_c} e^{-\lambda_{max}(X_j;A,\Theta)} + \sum_{j\in\Omega_e} e^{\lambda_{max}(X_j;A,\Theta)} + \\ \sum_{j\in\Omega_l} |\lambda_{max}(X_j;A,\Theta)| \end{aligned} \tag{4.5}$$

where λ_{max} is the maximum Lyapunov exponent estimated from the time series of an oscillator (for a fast implementation see [103]). Cells are also partitioned into mutually exclusive sets according to the qualitative behavior we want to impose on them after initial transients lapsed during the evolution of the array. Chaotic behavior is imposed on cells denoted by Ω_c convergence to one of the equilibria is imposed on cells denoted by Ω_e and limit cycle behavior on cells denoted by Ω_l.

4.4 Experiments

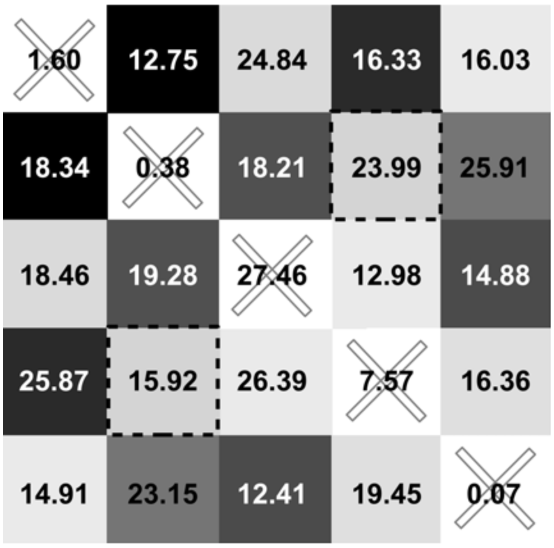
In all experiments presented, learning was performed as follows. Time evolution of the array was calculated using adaptive step-size solver ode45 in Matlab with relative error tolerance 10^{-3} on a time horizon $T = 600$. Time instants $t \in [450, 600]$ were used in calculating the cost value for each probed array configuration. Initial condition was set to the same value for all variables, with value 0.1 for all cells, except for one of the cells in the pair on which synchronization was imposed in the cost function where it was set to 0.5. The reason for such initial conditions was to ensure that the resulting synchronization is not due to the identical value of initial conditions.

Note that the relationship of behaviors found using our numerical simulation framework to the behavior occurring in the physical system is not clearly defined. However, the shadowing theorem [104] guarantees the existence of a true trajectory that remains close to the numerically produced trajectory (called the pseudo-trajectory) for very long times, i.e. the real system exhibits qualitatively the same behavior as our numerical simulations.

4.4.1 Cluster synchronization in arrays with unorganized interaction pattern

A new form of cluster synchronization is explored in cellular arrays of chaotic oscillators. Previously it was shown in the literature that symmetries of the coupling topology with uniform interaction weights lead to several coexisting clusters of synchronized cells. A new phenomenon is presented here where highly asymmetric interaction weights can give rise to cluster synchronization regimes with partial synchronization. In addition, cluster or partial synchronization regimes corresponding to asymmetric interaction patterns can limit the effect of the underlying symmetries of the network topology and boundary conditions at the expense of some residual synchronization error.

Ten mutually exclusive, coexisting cluster synchronization regimes can be observed in Fig. 4.3. Pairs of cells synchronize with respect to the principal diagonal of the square array but cells on the diagonal remain desynchronized. No straightforward spatial relationship can be noticed in the interaction pattern A that could



1.60	12.75	24.84	16.33	16.03
18.34	0.38	18.21	23.99	25.91
18.46	19.28	27.46	12.98	14.88
25.87	15.92	26.39	7.57	16.36
14.91	23.15	12.41	19.45	0.07

Figure 4.3: Ten mutually exclusive, coexisting cluster synchronization regimes in a 5 × 5 array of chaotic oscillators. Cells belonging to the same synchronizing cluster are marked with the same grayscale level. White cells marked with X do not synchronize to any other cell. Pairs of cells synchronize with respect to the principal diagonal of the array. However, no straightforward spatial relationship can be noticed in the interaction pattern A that could explain the symmetry in the spatial layout of cluster synchronization regimes. The clusters are not synchronized to each other despite the non-zero couplings between them. A cell is influenced by its nearest neighbors (four connected local topology) with coupling weight value $A_{r,c}$ shown in each cell, where index r denotes the row, c denotes the column of a cell.

explain the symmetry in the spatial layout of synchronizing cell clusters. The clusters are not synchronized to each other despite the non-zero couplings between them. This is a surprising phenomenon that may be related to the findings of [22] where a similar phenomenon was presented for an array where all coupling coefficients were identical. According to [22], topological products of synchronization regimes possible in a 1D chain of oscillators define all possible set of synchronization regimes in a 2D array. E.g. a possible regime is that cells synchronize in vertical stripes, i.e. a regime defined as the product of global synchronization in the vertical direction and global desynchronization in the horizontal direction. Cluster synchronization regimes in a 1D chain with zero flux boundary conditions

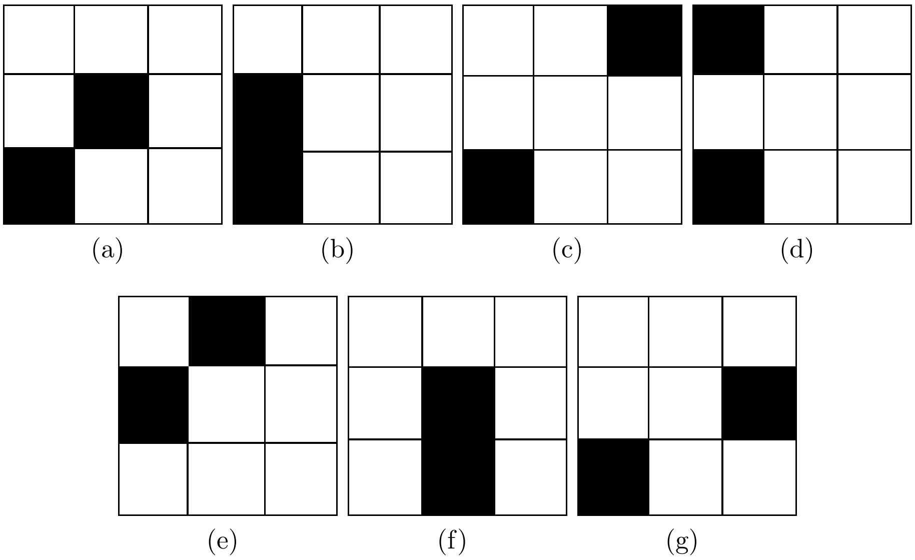
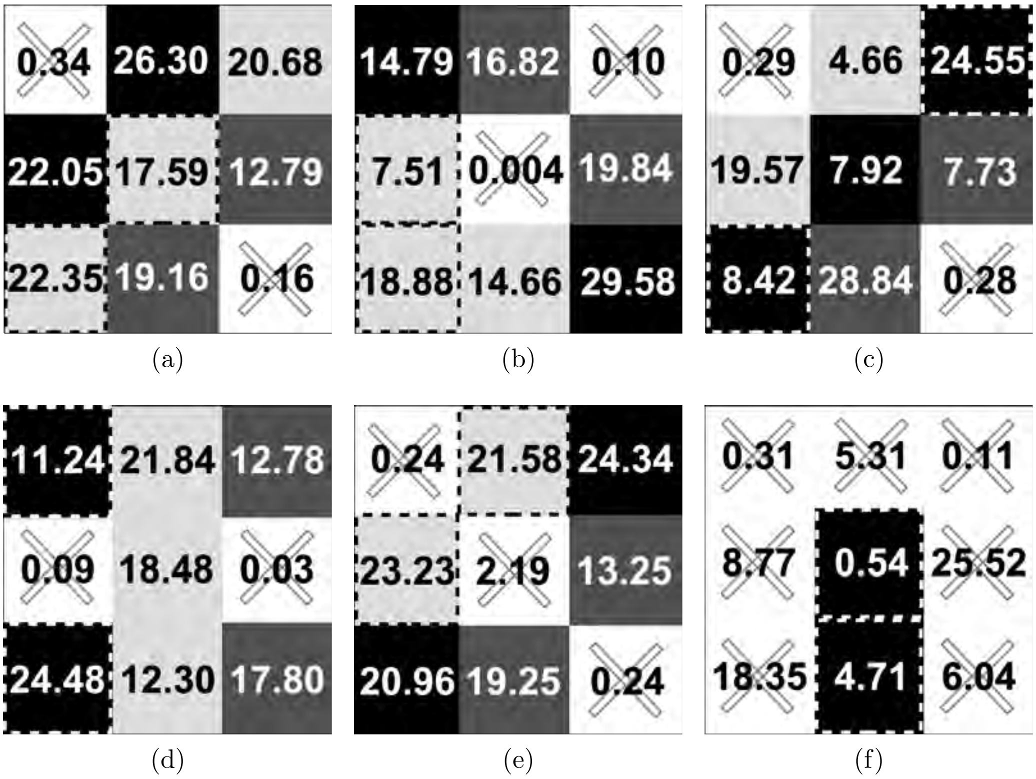


Figure 4.4: Stubs of possible cluster synchronization regimes in a 3 × 3 array, not including situations that can be obtained by rotating these stubs with 90°, 180° or 270°. Interaction patterns learned using our global optimization framework may synchronize either just these pairs or larger cell clusters that contain these stubs. All stubs may be part of a synchronization regime following an appropriate combination of axial symmetries as reported in [22]. The sole exception is the pair of cells shown in Fig. 4.4(g). These two cells can never synchronize in the same cluster if we assume the clusters must follow some combination of axial symmetries.

are symmetric to the middle of the chain. In a similar manner, in a 2D array, spatial layout of cluster synchronization regimes can arise from symmetries along the principal or secondary diagonal, the middle of the rows or columns or any combination of these axes. Putting no restriction on the interaction pattern, the possible spatial layouts of cluster synchronization regimes in a square array was investigated. Fig. 4.4 shows all stubs of possible cluster synchronization regimes in a 3 × 3 array, not including situations that can be obtained by rotating these stubs with 90°, 180° or 270°. Interaction patterns learned using our global optimization framework may synchronize either just these pairs or larger cell clusters that contain these stubs. Stubs of Fig. 4.4 may be part of a larger cluster synchronization regime following the spatial symmetry related rules outlined above.



(a) (b) (c) (d) (e) (f)

Figure 4.5: Six different cluster synchronization regimes learnt using Eq. (4.4). Cells belonging to the same synchronizing cluster are marked with the same grayscale level. White cells marked with X do not synchronize to any other cell. Dashed rectangles indicate the cells that were learnt to synchronize during optimization. A cell is influenced by its nearest neighbors (four connected local topology) with coupling weight value $A_{r,c}$ shown in each cell.

The sole exception is the pair of cells shown in Fig. 4.4(g) that is not symmetric to any axis.

Fig. 4.5 shows array configurations giving rise to cluster synchronization regimes that contain the specified stubs as specified in Fig.4.4(a) - 4.4(f). Also, in Fig. 4.3 the stub used in the learning process was symmetric with respect to the principal diagonal and it is contained in the cluster synchronization regime shown in Fig. 4.3. However, for the stub shown in Fig. 4.4(g), the only synchronization regime we found was when all cells in the array synchronized.

The cluster synchronization regimes shown in this section are persistent for random perturbations up to 20% of both the coupling and the bifurcation parameters. Persistence of the results were also confirmed for each experiment for a large number of random initial conditions.

The coupling matrix A was included in the learning process with inequality constraints $0 \leqslant A_{r,c} \leqslant 30$. Bifurcation parameters $\Theta_{r,c}$ were fixed, chosen from a normal distribution with mean 0.25 (the nominal value that produces the n-scroll attractor) and standard deviation 0.01. To obtain the configuration shown in Fig. 4.3 CSA with initial temperature 2.0 and 150 individual optimizers with 15000 population iterations was used. To obtain the configurations shown in Fig. 4.5 CSA with initial temperature 1.4 and 36 individual optimizers with 2500 population iterations was used.

4.4.2 Imposing qualitative behavior on cells

We performed learning for 2×2 and 3×3 on arrays of Chua's oscillators with hyperbolic tangent non-linearity, with eight-connected local topology, zero-flux boundary conditions. Both the coupling matrix and the matrix Θ of bifurcation parameters of each cell were included in the learning process. In Fig. 4.6 we show illustrative results for the 2×2 case where the target population to be synchronized was the bottom-left and upper-right cells. CSA with 16 individual optimizers and 200 population iterations ran for 20 minutes using ode45 with relative tolerance 10^{-3} on a Linux machine powered by an AMD Opteron 250. Similar qualitative effects can be observed for 3×3 arrays.

An important result of this study is that we showed that it is possible to learn partial synchronization regimes on arrays having different qualitative behavior. Figure 4.6 shows two examples for partial synchronization: chaotic dynamics emerges when limit cycle and convergent oscillators are coupled, but the same cells can synchronize also when coupling makes chaotic oscillators become limit-cycle oscillators.

4.4.3 Partial synchronization

Figure 4.7(a) and 4.7(b) show that the error between synchronizing cells is not zero. Such type of synchronization was already studied extensively in [98] where it was shown that qualitatively different dynamical systems can synchronize with some small residual error. Examples shown in this chapter confirm this phenomenon extending it into the more complex context of cluster synchronization occurring in networks defined in Eq. (4.1).

4.5 A link to living oscillator networks

Coupled oscillators are widely used to model neural dynamics in living organisms. One hot issue to be solved is to prove whether synchronization plays a role in processing information. A closely related, more fundamental issue is to determine the level of complexity of individual oscillators, i.e. what complexity is necessary to solve problems that biological systems solve in an efficient way. A widely adopted conjecture is that chaotic dynamics of individual oscillators is a necessary ingredient to solve tasks in the brain. However I only know of one published evidence that could confirm this conjecture. I give here some links between synchronization and neurobiology, and explore options for a framework that can be used to analyze the information processing capabilities of rectangular arrays of oscillators. The ultimate aim is to add application motivated aspects to existing results that so far focused on conditions for synchronization.

The binding problem first introduced by Rosenblatt et al. [105] is one of the most debated issues in neurobiology. It can be summarized as the task of understanding how the brain associates fragments of perceptual information belonging to the same object. These fragments can come from the same modality (e.g. Rosenblatt's example, see below) or from different ones like when one sees a stick hitting the ball and hears the sound of hitting and binds these sensory inputs to identify the single action of hitting the ball.

Rosenblatt's example can be seen in Fig. 4.8. A simple neural network is presented with a triangle or a square which can be in either the top or the bottom of a display. The network has four output neurons which fire to represent "square", "triangle", "top" and "bottom" respectively. Part B shows however that the network fails to separate cases when different inputs are present at the same time. This model cannot bind the information "triangle" to "top" and "square" to "bottom" when both parts of the figure has an input.

Although the brain usually does not appear to have a problem in correctly binding signals, as this simple example reveals, we still lack an understanding of how information variously distributed in patterns of neural firing results in coherent representations. In her review paper Treisman [106] decomposes the process of binding into three problems:

1. Parsing: what steps are involved in separating the single elements or events that need to be grouped from those not belonging to the object of interest. Attentional mechanisms are believed to play a role here, this falls not into the scope of our study.

2. Encoding : how is the binding encoded so that it can be signaled to other brain systems and used? Here, binding must have a role (see below).

3. Structural description: how are the correct relations specified between the bound elements within the same object? Here again, binding is a key issue to be understood in the process of how to connect objects after completing step 1 (see [106] for details).

Regarding the encoding problem, a number of mechanisms are believed to play a role, each having strong experimental cases for and against. One of the possible mechanisms is the binding by synchrony hypothesis proposed by Milner [107] and von der Malsburg [108].

Regarding the structural description problem, many studies confirming the role of synchrony in binding objects or events deduce their findings mostly from electroencephalography (EEG) data. In EEG, one measures voltage variations on an array of electrodes connected to the scalp. However each electrode reflects only an integrated version of the underlying activity of several neurons organized in cortical columns. Therefore synchronization phenomena observed in brain wave dynamics do not reveal how the underlying network processes information.

The fundamental aim of my investigations is to shed light on this last issue, i.e. to the question of how synchronization can play a role in information processing. My approach tries to exploit the advantage of being free from the issues common in neurobiology, like the inherent variability of cells that in itself makes the exploration of fundamental mechanisms very difficult.

In the engineering community, study of synchronization of chaotic oscillators has been a very active area for the last couple of years. Studies are focusing on deriving conditions for synchronization but we only have very limited knowledge why chaotic dynamics is beneficial in neural networks. Although these are fascinating questions from a purely scientific approach, it would be highly beneficial to justify the extra efforts needed to deal with such systems, i.e. having some clear application possibilities confirmed. So far only speculations and conjectures have come to surface suggesting that specific topological configurations and chaotic dynamics are advantageous without pointing out exactly why and in what situations chaotic dynamics is essential.

In [109] one step was made by showing how chaotic dynamics can solve the binding problem. It is known, that fixed-point attractors can only be used to retrieve one memory pattern at the same time. Multiple reinstantiation allows the retrieval of several memory patterns simultaneously, something that fixed-point attractors cannot realize. This is similar to the binding problem, since the multiple reinstantiation mechanism must connect multiple memory patterns during retrieval. Referring to Fig. 4.8, the recall process must connect square-up and triangle-down for the rightmost image. The solution could be synchronizing the firing pattern of the neurons sensitive for square with up and those of triangle with down. However, if two squares appear on the same figure, simple spike-train synchrony could not correctly interpret the situation: synchrony of both up and down with square would relate unrelated features (up and down are distinct, inherently non-overlapping features). This problem has a solution as show in Fig. 4.9. The gray spike trains form a so-called itinerant attractor, that is composed of attractor ruins. In the case depicted on Fig. 4.9, two ruins are present in the gray spike train from the upper and lower black spike trains. One ruin - that is a destabilized attractor - is connected to the other via dynamical, itinerant orbit. This way the property of being square is synchronized to both up and down without making those two mutually exclusive features collapse.

Given this constructive evidence (see details in [109]) for the role chaotic dynamics plays in pattern retrieval from associative memory sets, a number of questions remain unanswered.

One interesting issue is related to the so-called physiological moment, T, a short period of time ranging from 50 to 200 ms [110]. At time scale greater than T one can see a succession of conscious states. In the encoding phase (step 2) the binding problem must be solved at a time scale shorter than T since it is not a conscious process, however the structural description (step 3) goes above T. The difference between oscillators operating in limit cycle, chaotic or maybe even hyperchaotic mode could be reflected in the time needed to accomplish a specific task. From physiological data we have hints on the time scale a biological neurons processes information. Assuming we also know the length of the physiological moment, it would be interesting to show what kind of tasks can be accomplished using chaotic oscillator arrays in the given time-frames.

In [111] Belykh et al. showed for spiking neurons that only the number of inputs to a neuron is important as a condition for synchronization and not the network topology. In contrast, in [112] and in many other papers discussion is limited to non-spiking neurons. There seems to be a gap between papers dealing with spiking neuron models (e.g. Hindmarsh-Rose) and non-spiking ones (Chua, Lorenz, Rössler). As mentioned by János Neumann and also summarized in [113], the efficiency of the brain compared to computers comes from its optimized structure, i.e. it uses both analog and digital representations to process information. How is this related to the binding problem?

A number of models have been constructed to mimic the computational capabilities of the retina, known as essentially analog processing device. In [114] Bálya et al. qualitatively reproduced the computations the retina performs to extract the feature channels in response to a flashed square input. The model was implemented using a CNN-type analog Very Large Scale Integration (VLSI) processor, that represented a major step compared to the complexity and programmability to other models. Here, the fundamental question is whether there exists a rule of how to separate tasks that are more efficiently solved by non-spiking (purely analog) neurons (oscillators)? A solution to this question would be extremely revealing, also to circuit designers in motivating directions of their efforts.

A number of questions would be relevant to deal with:

1. Understand what does chaotic dynamics add to types of transformations a CNN can accomplish. Show examples where "classical" operators fail

and chaotic dynamics is essential (or it is more efficient). A candidate problem to solve is the binding problem. To be more specific, the question is whether the mechanisms revealed in [109] have a corresponding phenomenon in analog oscillators?

2. Recently conditions for cluster synchronization have been investigated by multiple studies. We know that cluster synchronization regimes are hyperplanes [112] (linear invariant manifolds). In the case of non-identical oscillators, these hyperplanes become non-linear manifolds. It was shown, that partial or cluster synchronization in diffusively coupled continuous chaotic oscillator networks is caused by symmetries in the network [112] [88]. However these results show only complete, asymptotic synchronization, that was not yet confirmed to play a role in biological systems. It seems to be too restrictive, as the work in [109] illustrates.

3. This underlines another issue, i.e. the lack of a good measure for synchronization. Although a number of methods appeared in the literature (see [115] for an overview) they do not solve the "accurate resolution - programmability - computationally effective" tri-requisite. Again, at the moment we just can admire how biological systems excel in solving the synchronization detection problem.

4. Find a good compromise between complexity and programmability, like it was the case for the retina model in [114]. Three candidate architectures can be named:

   (a) CNN array of Chua's oscillators. Programmable initial state vector, coupling matrix and parameter for the non-linearity $= 3+1+1 \times w \times h$ (w is the width of the array, i.e. the number of cells in horizontal dimension, h is the height of the array).

   (b) Multi-layer CNN. If we want to extend the already existing complex-cell VLSI CNN implementation [116], at least one layer should possess parameterizable, non-linear self-feedback. Implementing spatially invariant templates, this realization would need 2 time constants relative to the one fixed time-constant layer + 3 interlayer coupling coefficients + 3×9 intra-layer feed-back + 3×9 intra-layer feed-forward coefficients = 59 parameters for any array size.

(c) FPGA implementation of the first two possibilities. [72] reported an emulated digital implementation of a multi-layer CNN on FPGA. According to the results, only three CNN cores (three layer, nearest-neighbor interactions, 24 bit precision) could be implemented on the Xilinx XC2V8000. Despite some possible simplifications in the Chua's oscillator, FPGA implementation might not be a good candidate to boost simulations of chaotic CNNs.

The optimization framework used to study synchronization phenomena in this study - in the first time - was implemented as a computer simulation. I did not want to restrict possible synchronization regimes by symmetric coupling matrix and allowed different cells to be in different mode (limit-cyle, double scroll, multiple-scrolls). Therefore optimization techniques were used to search for array configurations that solve the desired tasks according to some measure.

Chaos and chaotic synchronization are believed to posses computational advantages but to my best knowledge there are only few evidences supporting this conjecture. A future goal can be to find further evidence and find the proper architecture that can serve as a hardware accelerator for this computationally expensive research topic.

4.6 Conclusions

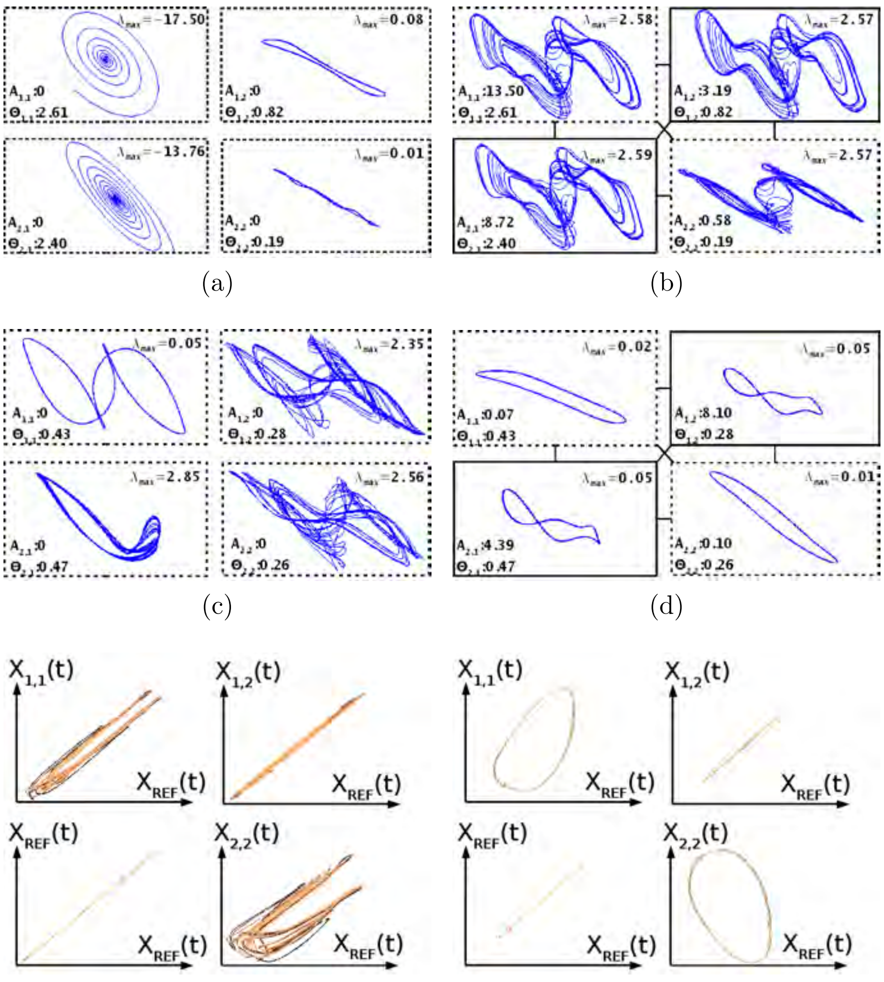
In [22] symmetries of the coupling topology were exploited to find cluster synchronization regimes of identical oscillators with identical coupling weights. In our study, the oscillators were not identical and the coupling coefficients were highly asymmetric, still the symmetries similar to those shown in [22] can be observed in the resulting cluster synchronization regimes. However, at the expense of some residual synchronization error, the asymmetry of the interaction pattern between cells can give rise to cluster synchronization regimes that constrain the effect of the underlying symmetries of the network topology and boundary conditions. This is a new phenomenon to be analyzed more in detail in future studies.

Global optimization techniques were successfully applied to find cluster synchronization regimes in chaotic oscillator arrays. A drawback of this approach is the high computational cost for large arrays that strongly motivates VLSI implementation of chaotic oscillator arrays. Hardware implementation could speed up experimentation and rule out the potential errors numerical simulations can introduce.

It was shown that with large enough number of iterations the global optimum is always reached using simulated annealing. However using optimization much caution is needed to make sure that the desired network behavior is properly translated into a cost function. Also, the fact that the desired behavior is not exhibited can be due to the insufficient number of optimization steps. The proposed synchronization metric has the advantage of requiring low computational effort compared to correlation based metrics.

The proposed metric for synchronization attributes $-\infty$ cost value to the complete synchronization of the selected cells, i.e. the difference between their state vectors becomes zero after some transition time interval but at the same time the difference between the remaining cells in the array is as high as possible. We think that the actual cost value can be used as an indicator that synchronization is happening, but this issue needs more rigorous investigations.

In future studies, it would be interesting to see what are the ingredients in a highly asymmetric coupling configuration that are essential in order to give rise to a specific cluster synchronization regime. A more involved question is to see how cluster synchronization is related to information processing in oscillator networks of living organisms.



(a)

(b)

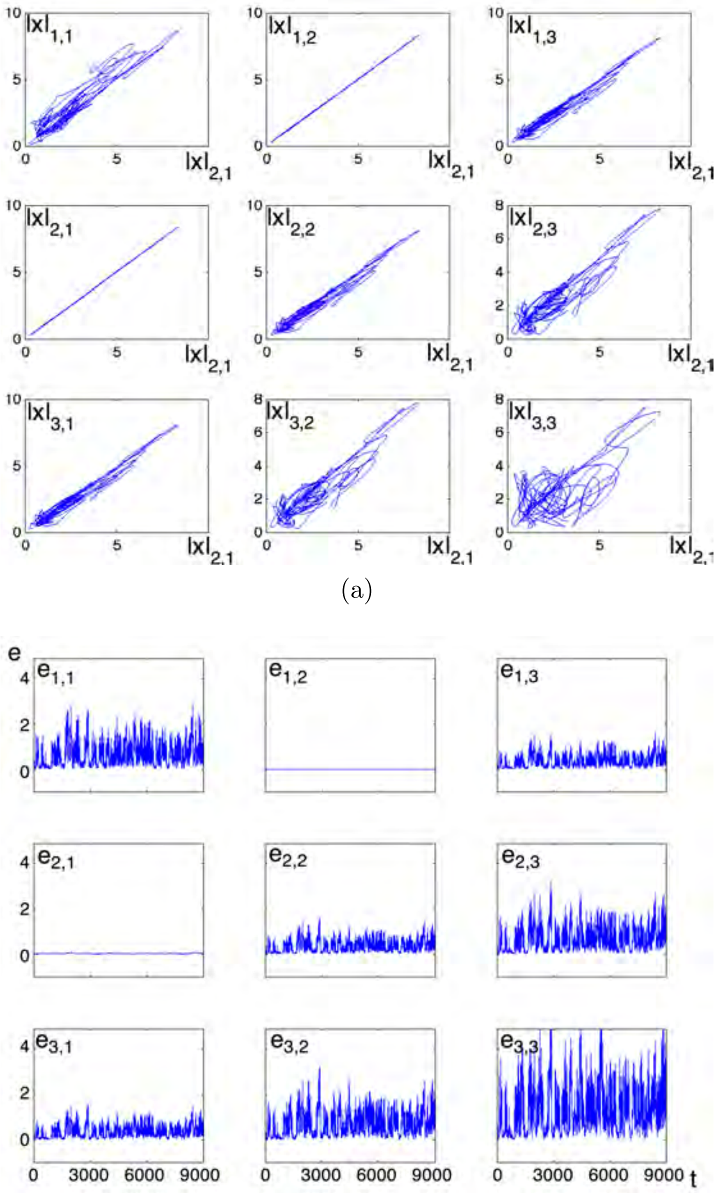
(c)

(d)

(e)

(f)

Figure 4.6: Trajectories showing the stationary behavior from time T to $2T$ of two qualitative behavior types of Chua's oscillator arrays. For both types, cells enclosed with solid lines were learned to synchronize. For a),b),e) positive maximum Lyapunov exponents (λ_{max}) were imposed on all cells. For c),d),f): zero λ_{max} were imposed on all cells. In a)-d) cells with continuous border are learnt to synchronize, dotted line enclosing denotes desynchronization. In a) and c) trajectories of the array without coupling are shown (no connecting lines between cells). b) and d) illustrate the behavior when coupling is added to array a) and c) respectively. e,f) Synchronization errors shown as $\|X_{REF}\|_2$ versus $\|X_{r,c}\|_2$ between a chosen reference cell and all other cells. Errors of array b) and d) are shown in e) and f) respectively. The coupling and bifurcation parameters $A_{r,c}$ and $\Theta_{r,c}$ are shown below each oscillator.



(a)

(b)

Figure 4.7: In Fig. 4.7(a) the magnitude of state vector of a cell is denoted by $|x|$. Magnitude of state vector of cell $(2, 1)$ was plotted against the same quantity of all other cells. Cells in the cluster marked with the darkest shade in Fig. 4.5(a) - i.e. cells $(2, 1)$ and $(1, 2)$ - synchronize as the 45° lines indicate. Fig. 4.7(b) shows the magnitude of the difference between the state vector of cell $(2, 1)$ and the others $e_{r,c} = \|x_{2,1} - x_{r,c}\|_2$. All synchronizing clusters in all experiments have qualitatively similar error plot.

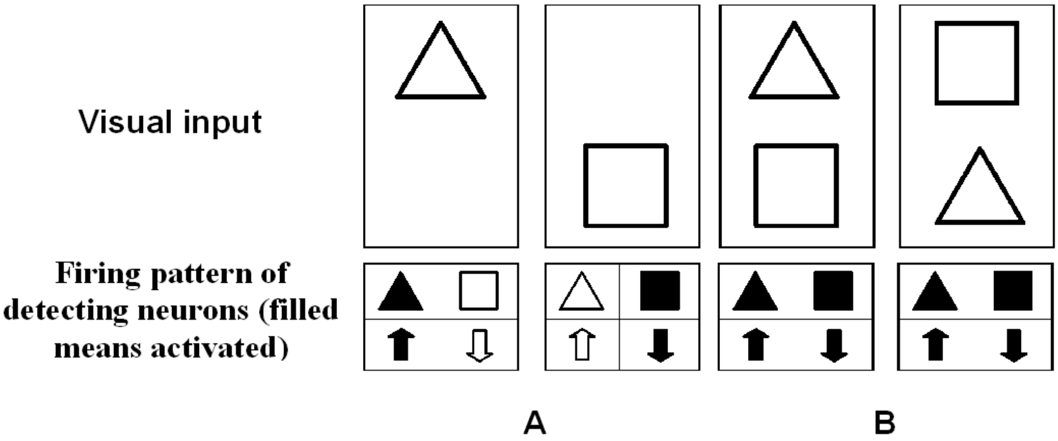

Visual input
Firing pattern of detecting neurons (filled means activated)
A
B

Figure 4.8: Rosenblatt's basic illustration to the binding problem. In the A case, the input can be coded with distinct neuron patterns. In case B, although the input is different, the activated neurons are the same, i.e. the network of neurons cannot distinguish between these cases. Taken from [105]

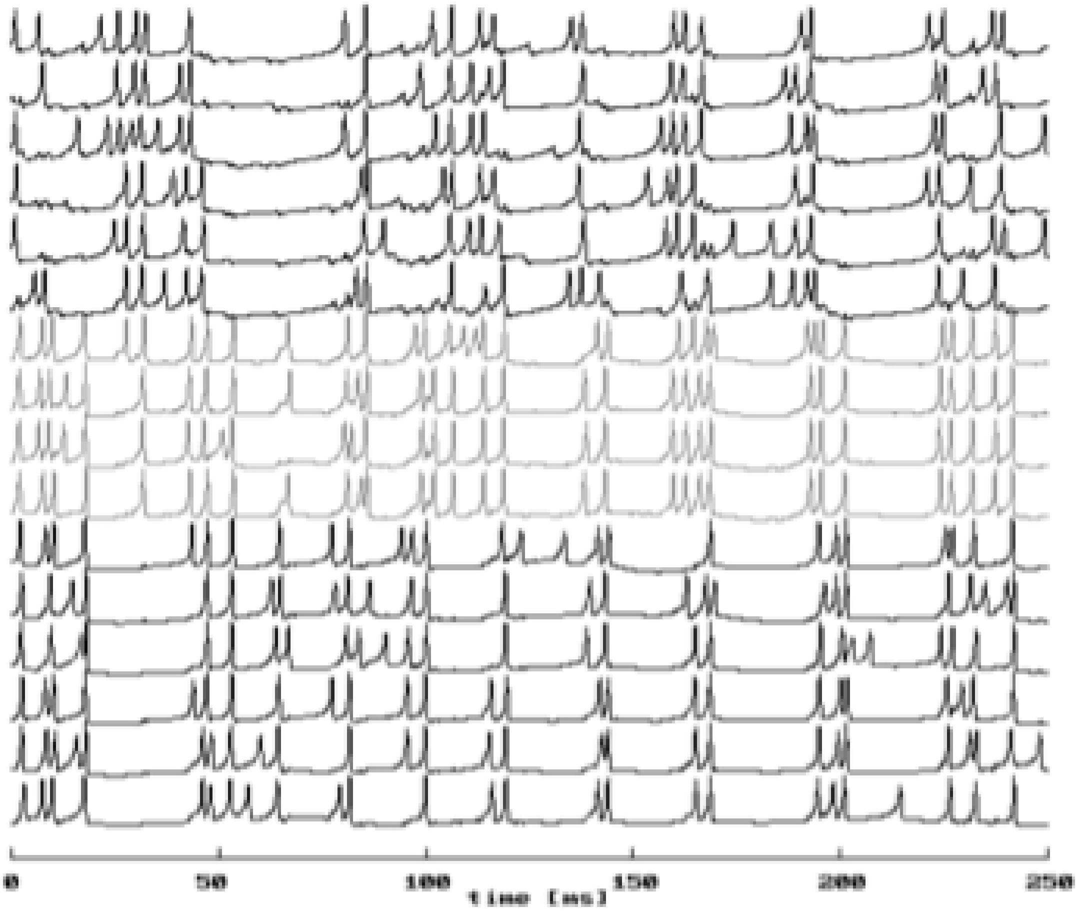


Figure 4.9: Itinerant synchronization of four neurons (the four gray spike-series in the middle) with two neuron populations. One population is formed by the first six lines with two grays included, the other population is the last six lines. Observe how the gray neurons synchronize alternating with the upper and lower six. The two populations are unsynchronized and compete for synchronization with the gray ones. Taken from [109].

Chapter 5

Application of the results

The results of Chapter 2 and Chapter 3 are in their current stage already very application oriented. Although the problems and the methods are different, at a higher abstraction level, these problems can be unified. A common lesson I learned is that the development of a good medical image processing method relies heavily on how much data is available of the target biological organ or feature and how strictly the data acquisition protocol is defined.

Results formulated in Chapter 4 are much more theoretical. I illustrated with case studies that my proposed approach can be a useful tool to test hypotheses made on coupled networks of chaotic oscillators. Rules (either theoretical or empirical) believed as generic can be transformed into a cost function. Then a global optimization process can maximize the cost function, i.e. it looks for solutions that violate the rule formulated as a cost function. Finding no such solution can be a strong evidence supporting the validity of the original rule.

Confirming generic (theoretical) rules via global optimization might seem improper. However, deriving sound theoretical results is extremely difficult when dealing with coupled networks of chaotic oscillators. Therefore the method showed in this dissertation can provide valuable feedback. This study showed that by using a global optimization framework it is possible to get insight into the complex interactions of coupled chaotic oscillators. Our method can be useful in guiding investigations and confirming results from theoretical analysis.

My motivation for investigating synchronization phenomena stemmed from the insight I gained into high speed image processing problems. I realized that despite the very good operation per joule figure of CNN processors, performance of machine vision algorithms is still very low compared to the ease of living creatures tackling complicated scenarios. Actually, we still lack the understanding of what mechanisms should be exploited to construct high speed, highly robust, low power vision algorithms. On the other hand, in neurobiology, the same problem arises when trying to understand how neurons actually perform computation.

In the engineering community, study of synchronization of chaotic oscillators became a very active area since a couple of years. Studies are focusing on deriving conditions for synchronization but we only have very limited knowledge why chaotic dynamics is beneficial in neural networks. So far only speculations and conjectures have come to surface suggesting that specific topological configurations and chaotic dynamics are advantageous without pointing out exactly why and in what situations chaotic dynamics is essential. Although synchronization and chaotic dynamics are fascinating topics from a purely scientific approach, it would be highly beneficial to justify the extra efforts needed to deal with such systems. My method could represent a step in this direction.

Appendix A

Appendix

The appendix aims to provide the reader with details on the implementation of the routines used in the pseudocodes of Chapter 2 and Chapter 3. Widely known CNN templates and Matlab commands are however not discussed here and the interested reader is referred to [117][118][119]. Details on CNN implementation aspects were added where necessary.

A.1 General terms, notations and conventions

If not stated otherwise the following notation is used in operator, subroutine and algorithm descriptions:

- I - denotes an image (2D array of pixel values)
- $*$ - denotes convolution
- k - time index
- i, j - spatial indices
- σ - spatial convolution parameter
- ϕ - spatial directional parameter
- θ - threshold parameter

A.2 Basic operators

Threshold – thresholds a gray-scale input image at a given gray-scale level. The output is a binary image defined as follows:

$$Thr(I(i,j),\theta) = \begin{cases} 1 & if\ I(i,j) > \theta \\ 0 & otherwise \end{cases}$$

CNN implementation: by using the Thresh template.
ACE16k implementation: available (stable).

Normalize - calculates a normalized version of a gray-scale input image. The formulation of the operation is as follows (D_{min} and D_{max} stand for the minimum and maximum of the available dynamic range; I_{min} and I_{max} stand for the minimum and maximum of the input image, respectively):

$$Norm(I(i,j)) = D_{min} + \frac{D_{max} - D_{min}}{I_{max} - I_{min}}(I(i,j) - I_{min}) = D_{min} + \frac{\Delta D}{\Delta I}(I(i,j) - I_{min})$$

CNN implementation: first I_{min} and I_{max} is calculated by using the Thresh template and global logic. Since ΔD is known a priori, the implementation of the above formula leads to an analogic algorithm based on template Scale and simple arithmetics. Remark: the constant $b_0 = \Delta D/\Delta I$ that is the central element of the B term in Scale is image dependent, thus interaction with the digital environment is needed.
ACE16k implementation: -.

Diffuse - calculates a linear low-pass filtered version of a gray-scale input image. The formulation of the operation is as follows:

$$Diffus(I,\sigma) = I * G(\sigma)$$

CNN implementation: the above equation describes a linear convolution by a Gaussian kernel. Under fairly mild conditions at some time t this corresponds to the solution of a diffusion type partial differential equation. After spatial discretization, the PDE can be mapped to a CNN structure programmed by template Diffus. In this form the the transient length is explicitly related to $G(t \approx \sqrt{\sigma})$.

ACE16k implementation: iterated convolution - available (stable), continuous diffusion - available (resistive grid solution - stable).

Dir_Gradient - calculates the directional gradient along the specified cardinal direction. Directionality is represented by the actual mask image used for the operation. CNN implementation: by using the direction specific variants of the Sobel template as follows (example is given for the North direction):

$$B = \begin{pmatrix} 0.25 & 0.5 & 0.25 \\ 0 & 0 & 0 \\ -0.25 & -0.5 & -0.25 \end{pmatrix} \; z = -0.001$$

ACE16k implementation: available (stable).

A.3 Subroutines

A.3.1 Subroutines for the CWC algorithm

Trigger Wave Dilation - dynamic dilation of a wave-front initiated from patch-like sources. The I_{Mark} image contains the marker image that contains the seed locations of reconstruction. The mask image I_{Mask} contains binary constraints to the reconstruction. τ is the number of iterations allowed (temporal constraint). Trigger Wave Erosion works in a similar manner but the role of black and white colors are interchanged.
CNN implementation: A linear CNN template $T = [a_0 \ \beta \ d]$ is a trigger-wave generator if $a_0 > 1$ and $\theta_L < d < \theta_H$ is satisfied; where a_0 is the central element and β are the surrounding elements of the feed-back template, d represents the control term with $\theta_L = 1 - \alpha - N_1 \cdot \beta$ and $\theta_H = -1 + \alpha + N_1 \cdot \beta$. Typical values for trigger wave generator are $T = [3 \ 0.25 \ 3.75]$. A contains the elements of the feedback matrix. Properly changing the values of the A template can make the propagation anisotropic. The value of d controls the wave between expanding and shrinking mode. The PATCHM template performs binary mask constrained wave propagation.

```
function TrW_Dilation($I_{Mark}, I_{Mask}, \tau$)
    $I_{Core} = I_{Mark}$
    for $iter = 1 : \tau$ do
        for $i = 1 : M$ do
            for $j = 1 : N$ do
                $c = I_{Core}(i, j)$
                if $c == BLACK$ then
                    $I_{CWP}(i, j) = BLACK$
                    continue
                end if
                $n = CountNeighbors(c)$
                if $n < 2$ then
                    $I_{CWP}(i, j) = WHITE$
                    continue
                end if
                if $I_{Mask}(i, j) == WHITE$ then
                    $I_{CWP}(i, j) = WHITE$
                    continue
                end if
                $I_{CWP}(i, j) = BLACK$
            end for
        end for
        $I_{Core} = I_{CWP}$
    end for
end function
```

ACE16k implementation: available (stable).

Constrained Trigger Wave Dilation - calculates the steady contour of a wave-front initiated from patch-like sources. An evolving wave-front can be controlled through a grayscale valued spatial constraint, i.e. by the input image I_{GS}. The I_{Mark} image contains the marker image that contains the seed locations of reconstruction. The mask image I_{Mask} contains binary constraints to the reconstruction. θ is a grayscale threshold that identifies regions in I_{GS} where the propagation should be stopped. τ is the number of iterations allowed (temporal constraint). Constrained Trigger Wave Erosion works in a similar manner but the role of black and white colors are interchanged.
CNN implementation: The PATCHMC template performs grayscale constrained trigger wave propagation. The difference between PATCHM and PATCHMC is the presence of the central element in the B matrix, i.e. in PATCHMC, b_0 is the central element of the feed-forward matrix that weighs the input mask I_{GS}.
ACE16k implementation: available (stable).

MotionEstimation - calculates a smooth motion intensity map. Smoothness is ensured by applying a diffusion operator on the input images. This could be interpreted as the application of a temporal DoG (difference of Gaussians) operator. If the target image feature moves a lot between two frames, $I_{preprocC}$ should be used instead of $I_{preprocN}$. The use of the "next" or read-ahead frame is justified when the time resolution of the input flow is fine. In such cases, the wall motion constituent of the spatial constraint may be too small.

CalcBWMask - calculates a BW mask that poses a hard limit on trigger wave propagation. $I_{Fixmask}$ is a binary image where white pixels cover the region that falls outside the ultrasound beam, i.e. outside of the apriori known region of interest.

Combine - normalizes and calculates the weighted sum of two input images.

```
function TrW_Dilation($I_{Mark}, I_{Mask}, \tau$)
    $I_{Core} = I_{Mark}$
    for $iter = 1 : \tau$ do
        for $i = 1 : M$ do
            for $j = 1 : N$ do
                $c = I_{Core}(i, j)$
                if $c == BLACK$ then
                    $I_{CWP}(i, j) = BLACK$
                    continue
                end if
                $n = CountNeighbors(c)$
                if $n < 2$ then
                    $I_{CWP}(i, j) = WHITE$
                    continue
                end if
                if $I_{Mask}(i, j) == WHITE$ then
                    $I_{CWP}(i, j) = WHITE$
                    continue
                end if
                $s = AverNeighbors(I_{GS}(i, j))$
                if $s < \theta$ then
                    $I_{CWP}(i, j) = BLACK$
                else
                    $I_{CWP}(i, j) = WHITE$
                end if
            end for
        end for
        $I_{Core} = I_{CWP}$
    end for
end function
```

```
function MotionEstimation($I_{preprocN}, I_{preprocP}, \sigma_1, \sigma_2$)
    $I_{DN} = Diffus(I_{preprocN}, \sigma_1)$
    $I_{DP} = Diffus(I_{preprocP}, \sigma_2)$
    $I_{MIM} = I_{DN} - I_{DP}$
end function
```

```
function CalcBWMask($I_{PrevResult}, I_{Fixmask}, t_d$)
    $I_D = TrW\_Dilation(I_{PrevResult}, t_d)$
    $I_{BWMask} = I_D \ AND \ I_{Fixmask}$
end function
```

```
function COMBINE($I_1, I_2, \mathbf{w}$)
    $I_{1n} = Norm(I_1)$
    $I_{2n} = Norm(I_2)$
    $I_{Out} = I_{1n} * \mathbf{w}(1) + I_{2n} * \mathbf{w}(2)$
end function
```

A.3.2 Subroutines for Pixel Level Snakes

Balloon Force Estimation - returns an internal inflating potential where input is the contour image, w gives the real potential value weight, cf is a flag that switches between compresion/dilation potential (1 = compression) and the output is the inflating potential image.

```
function BALLOONFORCEEST($I_{in}, w, cf$)
    $I_P = HoleFilling(I_{in})$
    if $cf == 1$ then
        $I_P = NOT(I_{in})\ AND\ I_P$
        $I_P = NOT(I_P)$
    end if
    $I_{Out} = I_P * w$
end function
```

Directional Contour Expansion (DCE) - Performs a directional expansion of the contour along the desired direction driven by binary external information.

```
function DCE($dir, I_{AC}, I_GFF$)
    $I_{Shifted} = Shift(I_{AC}, dir)$
    $I_{Exp} = I_{Shifted}\ AND\ I_{AC}$
    $I_{Out} = I_{Exp}\ OR\ I_{AC}$
end function
```

Directional Contour Thinning (DCT) - Performs a directional thinning of the contour along the desired direction driven by binary external information. Directionality is represented by the actual mask image used for the operation.

```
function DCT($I_{AC}, I_{Mask}$)
    $I_{Exp} = HitAndMiss(I_{AC}, I_{Mask})$
    $I_{Out} = NOT\ I_{Exp}\ AND\ I_{AC}$
end function
```

Guiding Force Estimation (GFE) - Performs a ”binary” directional gradient on the potential image along the desired direction. The output is a binary map with black pixels where the guiding forces are positive along the direction being processed.

```
function GFE($dir, I_{Ext}, I_{Int}, I_{Balloon}$)
    $I_{SumPot} = I_{Ext} + I_{Int} + I_{Balloon}$
    $I_{GF} = Dir\_Gradient(dir, I_{SumPot}$
    $I_{Out} = Threshold(I_{GF}, 0)$
end function
```

Internal Potential Estimation (IPE) - Performs the internal potential estimation for the current active contour.

```
function IPE($I_{AC}, iter, w$)
    $I_{Diff} = Diffuse(I_{AC}, iter)$
    $I_{Out} = I_{Diff} * w$
end function
```

A.3.3 Subroutine descriptions for retinal ganglion cell classification

$yy = smooth(y)$ smooths the data in the vector y using the LOESS method.

$[pospeakind, negpeakind] = peakdetect(signal)$ generates the positive/negative polarity (concave down/convex) peak index vectors from the *signal* vector.

$[ind] = crossing(S, level, slope)$ returns an index vector ind containing the indexes where the signal S crosses the value set in *level*. Specifying ’ascent’ or 'descent' in *slope* can restrict the search to rising or falling slopes.

$[counts, x] = hist(I)$ returns the histogram counts in *counts* and the bin locations in x. For indexed images, hist returns the histogram counts for each colormap entry; the length of counts is the same as the length of the colormap.

$BW2 = bwmorph(BW, operation)$ applies a specific morphological operation to the binary image BW.
Erosion: take the 3 × 3 neighborhood of each pixel in BW. Set a pixel white if at least one of its neighbors is white.
Dilation: set a pixel black if at least one of its neighbors is black.
Opening: erosion followed by dilation.
Closing: dilation followed by erosion.
Skeleton: the skeleton of a filled black object consists of the set of points that are equally distant from two closest points of the object's boundary.

$IM = imreconstruct(marker, mask)$ performs morphological reconstruction of the image marker under the image mask. marker and mask can be two intensity images or two binary images with the same size. The returned image IM is an intensity or binary image, respectively. marker must be the same size as mask, and its elements must be less than or equal to the corresponding elements of mask.

$L = bwlabel(BW, n)$ returns a matrix L, of the same size as BW, containing labels for the connected objects in BW. The elements of L are integer values greater than or equal to 0. The pixels labeled 0 are the background. The pixels labeled 1 make up one object, the pixels labeled 2 make up a second object, and so on.

$STATS = regionprops(L, properties)$ measures a set of properties for each labeled region in the label matrix L. Positive integer elements of L correspond to different regions. For example, the set of elements of L equal to 1 corresponds to region 1; the set of elements of L equal to 2 corresponds to region 2; and so on. The return value $STATS$ is a structure array of length $max(L(:))$. The fields of

the structure array denote different measurements for each region, as specified by properties.
Eccentricity: a scalar corresponding to the eccentricity of the ellipse that has the same second-moments as the region. The eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length. The value is between 0 and 1, 0 and 1 are degenerate cases; an ellipse whose eccentricity is 0 is actually a circle, while an ellipse whose eccentricity is 1 is a line segment.
Centroid: the ratios in every spatial direction of the first and zero order moments of the object define its centroid
BoundingBox: the smallest rectangle that encompasses the object and is oriented along the major axis of the object.

$regionalmaxima(I)$: points in the image where all neighbor pixel value is smaller.

$fitwin(centers, width)$: returns the upper left coordinates of square windows with specified $width$ and $centers$.

cat: concatenate.

References

The author's journal publications

[1] D. Hillier, V. Binzberger, D. L. Vilarino, and C. Rekeczky, "Topographic cellular active contour techniques: Theory, implementations and comparisons," *International Journal of Circuit Theory and Applications*, vol. 34, no. 2, pp. 183–216, 2006.

[2] T. Viney, K. Balint, D. Hillier, S. Siegert, Z. Boldogkoi, L. Enquist, M. Meister, C. Cepko, and B. Roska, "Local Retinal Circuits of Melanopsin-Containing Ganglion Cells Identified by Transsynaptic Viral Tracing.," *Curr Biol*, vol. 17, pp. 981–988, 2007.

[3] D. Hillier, S. Günel, J. A. K. Suykens, and J. Vandewalle, "Partial synchronization in oscillator arrays with asymmetric coupling," *International Journal of Bifurcation and Chaos*, vol. 17, no. 11, 2007.

The author's international conference publications

[4] D. Hillier, S. Xavier-de Souza, J. Suykens, and J. Vandewalle, "CNNOPT: Learning dynamics and CNN chip-specific robustness," *IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA)*, 2006.

[5] Z. Szálka, G. Soós, D. Hillier, L. Kék, G. Andrássy, and C. Rekeczky, "Space-time signature analysis of 2D echocardiograms based on topographic cellular active contour techniques," *IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA)*, 2006.

[6] D. Hillier, Z. Czeilinger, and C. Rekeczky, "On-line 3D reconstruction of cardiac chambers via topographic cellular algorithms from echocardiographic sequences," *International Symposium on Nonlinear Theory and its Applications (NOLTA)*, 2005.

[7] D. Hillier, V. Binzberger, D. L. Vilarino, and C. Rekeczky, "Topographic cellular active contour techniques: Comparison of different approaches," *IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA)*, 2004.

[8] C. Rekeczky, V. Binzberger, D. Hillier, G. Soós, G. Cserey, and D. L. Vilarino, "A diagnostic echocardiography system booster by CNN technology," *IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA)*, 2004.

[9] D. Hillier and B. Roska, "Fully automated analysis of neural morphologies in the mammalian retina," *Annual meeting of the Friedrich Miescher Institute, Pontresina*, 2005.

[10] D. Hillier, S. Günel, J. A. K. Suykens, and J. Vandewalle, "Learning partial synchronization regimes with imposed qualitative behavior on an array of chua's oscillators," *Proc. Int. Symp. Nonlinear Theory and its Applications (NOLTA)*, pp. 983–986, 2006.

[11] D. Hillier, S. Günel, J. A. K. Suykens, and J. Vandewalle, "Partial synchronization in oscillator arrays with asymmetric coupling," *Proc. Int. Symp. on Synchronization in Complex Networks, Leuven*, 2007.

Publications on topics connected to the dissertation

[12] L. O. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Trans. on Circuits and Systems*, vol. 35, pp. 1257–1272, 1988.

[13] L. O. Chua and L. Yang, "Cellular neural networks: Applications," *IEEE Trans. on Circuits and Systems*, vol. 35, pp. 1273–1290, 1988.

[14] H. Harrer and J. A. Nossek, "Discrete-time cellular neural networks," *International Journal of Circuit Theory and Applications*, vol. 20, pp. 453–468, 1992.

[15] L. O. Chua and T. Roska, "The CNN paradigm," *IEEE Trans. on Circuits and Systems*, vol. 40, pp. 147–156, 1993.

[16] T. Roska and L. O. Chua, "The CNN universal machine: an analogic array computer," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on [see also Circuits and Systems II: Express Briefs, IEEE Transactions on]*, vol. 40, no. 3, pp. 163 – 173, 1993.

[17] T. Roska, "Dual computing structures containing analog cellular neural networks and digital decision units," *in Proc. IFIP Workshop on Silicon Architectures for Neural Nets*, pp. 233–244, 1991.

[18] L. O. Chua, T. Roska, and P. L. Venetianer, "The CNN is as universal as the Turing machine," *IEEE Trans. on Circuits and Systems*, vol. 40, pp. 289–291, 1993.

[19] G. Linan, S. Espejo, R. Dominguez-Castro, and A. Rodriguez-Vazquez, "ACE 4 k: An analog I/O 64× 64 visual microprocessor chip with 7-bit analog accuracy," *International Journal of Circuit Theory and Applications*, vol. 30, no. 2-3, pp. 89 – 116, 2002.

[20] A. Rodriguez-Vazquez, G. Linan-Cembrano, L. Carranza, E. Roca-Moreno, R. Carmona-Galan, F. Jimenez-Garrido, R. Dominguez-Castro, and S. E. Meana, "ACE16k: the third generation of mixed-signal SIMD-CNN ACE chips toward VSoCs," *Circuits and Systems I: Regular Papers, IEEE Transactions on [see also Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on]*, vol. 51, no. 5, pp. 851 – 863, 2004.

[21] Z. Liu, Y.-C. Lai, and M. A. Matías, "Universal scaling of Lyapunov exponents in coupled chaotic oscillators," *Physical Review E*, vol. 67, no. 045203, 2003.

[22] I. Belykh, V. Belykh, K. Nevidin, and M. Hasler, "Persistent clusters in lattices of coupled nonidentical chaotic systems," *Chaos*, vol. 13, pp. 165–178, 2003.

[23] B. J. Krenning, M. M. Voormolen, and J. R. T. C. Roelandt, "Assessment of left ventricular function by three-dimensional echocardiography," *Cardiovascular Ultrasound*, vol. 1, p. 12, 2003.

[24] A. F. Frangi, W. J. Niessen, and M. A. Viergever, "Three-dimensional modeling for functional analysis of cardiacimages, a review," *Medical Imaging, IEEE Transactions on*, vol. 20, no. 1, pp. 2 – 5, 2001.

[25] J. W. Klingler, C. L. Vaughan, T. Fraker, and L. Andrews, "Segmentation of echocardiographic images using mathematical morphology," *Biomedical Engineering, IEEE Transactions on*, vol. 35, no. 11, pp. 925–934, 1988.

[26] J. M. B. Dias and J. M. N. Leitao, "Wall position and thickness estimation from sequences of echocardiographic images," *IEEE Transactions on Medical Imaging*, vol. 15, no. 1, pp. 25 – 38, 1996.

[27] S. K. Setarehdan and J. J. Soraghan, "Automatic Cardiac LV Boundary Detection and Tracking Using Hybrid Fuzzy Temporal and Fuzzy Multiscale Edge Detection," *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, vol. 46, no. 11, 1999.

[28] C. Kotropoulos, X. Magnisalis, I. Pitas, and M. G. Strintzis, "Nonlinear ultrasonic image processing based on signal-adaptivefilters and self-organizing neural networks," *Image Processing, IEEE Transactions on*, vol. 3, no. 1, pp. 65 – 77, 1994.

[29] V. Chalana, D. T. Linker, and D. R. Haynor, "A multiple active contour model for cardiac boundary detection onechocardiographic sequences," *Medical Imaging, IEEE Transactions on*, vol. 15, no. 3, pp. 290 – 298, 1996.

[30] J. G. Bosch, S. C. Mitchell, B. P. F. Lelieveldt, F. Nijland, O. Kamp, M. Sonka, and J. H. C. Reiber, “Automatic segmentation of echocardiographic sequences by active appearance motion models,” *Medical Imaging, IEEE Transactions on*, vol. 21, no. 11, pp. 1374 – 1383, 2002.

[31] S. C. Mitchell, J. G. Bosch, B. P. F. Lelieveldt, R. J. van der Geest, J. H. C. Reiber, and M. Sonka, “3-D active appearance models: segmentation of cardiac MR and ultrasound images,” *Medical Imaging, IEEE Transactions on*, vol. 21, no. 9, pp. 1167 – 1178, 2002.

[32] R. Beichel, H. Bischof, F. Leberl, and M. Sonka, “Robust active appearance models and their application to medical image analysis,” *Medical Imaging, IEEE Transactions on*, vol. 24, no. 9, pp. 1151 – 1169, 2005.

[33] C. Corsi, G. Saracino, A. Sarti, and C. Lamberti, “Left ventricular volume estimation for real-time three-dimensional echocardiography,” *Medical Imaging, IEEE Transactions on*, vol. 21, no. 9, pp. 1202 – 1208, 2002.

[34] N. Paragios, “A level set approach for shape-driven segmentation and tracking of the left ventricle,” *Medical Imaging, IEEE Transactions on*, vol. 22, no. 6, pp. 773 – 776, 2003.

[35] J. Yang, L. H. Staib, and J. S. Duncan, “Neighbor-constrained segmentation with level set based 3-D deformable models,” *Medical Imaging, IEEE Transactions on*, vol. 23, no. 8, pp. 940 – 948, 2004.

[36] I. Wolf, M. Hastenteufel, R. D. Simone, M. Vetter, G. Glombitza, S. Mottl-Link, C. F. Vahl, and H. P. Meinzer, “ROPES: a semiautomated segmentation method for accelerated analysis of three-dimensional echocardiographic data,” *Medical Imaging, IEEE Transactions on*, vol. 21, no. 9, pp. 1091 – 1104, 2002.

[37] P. Abolmaesumi and M. R. Sirouspour, “An interacting multiple model probabilistic data association filter for cavity boundary extraction from ultrasound images,” *Medical Imaging, IEEE Transactions on*, vol. 23, no. 6, pp. 772 – 784, 2004.

[38] M. Song, R. M. Haralick, F. H. Sheehan, and R. K. Johnson, "Integrated surface model optimization for freehand three-dimensional echocardiography," *Medical Imaging, IEEE Transactions on*, vol. 21, no. 9, pp. 1077 – 1090, 2002.

[39] C. Rekeczky, A. Tahy, Z. Vegh, and T. Roska, "CNN-based spatio-temporal nonlinear filtering and endocardial boundary detection in echocardiography," *International Journal of Circuit Theory and Applications*, vol. 27, no. 1, pp. 171 – 207, 1999.

[40] C. Rekeczky and L. O. Chua, "Computing with Front Propagation: Active Contour And Skeleton Models In Continuous-Time CNN," *The Journal of VLSI Signal Processing*, vol. 23, no. 2, pp. 373 – 402, 1999.

[41] J. Montagnat, H. Delingette, and G. Malandain, "Cylindrical Echocardiographic Images Segmentation based on 3D Deformable Models," *Medical Image Computing and Computer-Assisted Intervention (MICCAI'99)*, vol. 1679, 1999.

[42] J. Montagnat, M. Sermesant, H. Delingette, G. Malandain, and N. Ayache, "Anisotropic filtering for model-based segmentation of 4D cylindrical echocardiographic images," *Pattern Recognition Letters*, vol. 24, no. 4-5, pp. 815 – 828, 2003.

[43] E. D. Angelini, A. F. Laine, S. Takuma, J. W. Holmes, and S. Homma, "LV volume quantification via spatiotemporal analysis of real-time 3-D echocardiography," *Medical Imaging, IEEE Transactions on*, vol. 20, no. 6, pp. 457 – 469, 2001.

[44] O. Gérard, A. C. Billon, J. M. Rouet, M. Jacob, M. Fradkin, and C. Allouche, "Efficient Model-Based Quantification of Left Ventricular Function in 3-D Echocardiography," *IEEE TRANSACTIONS ON MEDICAL IMAGING*, vol. 21, no. 9, p. 1059, 2002.

[45] V. Zagrodsky, V. Walimbe, C. R. Castro-Pareja, J. X. Qin, J. M. Song, and R. Shekhar, "Registration-Assisted Segmentation of Real-Time 3-D

Echocardiographic Data Using Deformable Models," *IEEE TRANSACTIONS ON MEDICAL IMAGING*, vol. 24, no. 9, p. 1089, 2005.

[46] G. Jacob, J. A. Noble, C. Behrenbruch, A. D. Kelion, and A. P. Banning, "A shape-space-based approach to tracking myocardial borders andquantifying regional left-ventricular function applied inechocardiography," *Medical Imaging, IEEE Transactions on*, vol. 21, no. 3, pp. 226 – 238, 2002.

[47] G. I. Sanchez-Ortiz, G. J. T. Wright, N. Clarke, J. Declerck, A. P. Banning, and J. A. Noble, "Automated 3-D echocardiography analysis compared with manual delineations and SPECT MUGA," *Medical Imaging, IEEE Transactions on*, vol. 21, no. 9, pp. 1069 – 1076, 2002.

[48] D. Comaniciu, X. S. Zhou, and S. Krishnan, "Robust real-time myocardial border tracking for echocardiography: an information fusion approach," *Medical Imaging, IEEE Transactions on*, vol. 23, no. 7, pp. 849 – 860, 2004.

[49] I. Dindoyal, T. Lambrou, J. Deng, C. Ruff, A. Linney, C. Rodeck, and A. Todd-Pokropek, *Functional Imaging and Modeling of the Heart*, ch. Level Set Segmentation of the Fetal Heart, pp. 123–132. Springer, 2005.

[50] A. Sarti, R. Malladi, and J. Sethian, "Subjective Surfaces: A Geometric Model for Boundary Completion," *International Journal of Computer Vision*, vol. 46, no. 3, pp. 201–221, 2002.

[51] R. Courant and F. John, *Introduction to Calculus and Analysis: Vol.: 2*. Springer, 1989.

[52] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321 – 331, 1988.

[53] L. Cohen and I. Cohen, "Finite-element methods for active contour models and balloons for 2-D and 3-D images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1131–1147, 1993.

[54] A. Blake and M. Isard, *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion.* Springer-Verlag New York, Inc. Secaucus, NJ, USA, 1998.

[55] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[56] R. Malladi, J. Sethian, and B. Vemuri, "Shape modeling with front propagation: a level set approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 158–175, 1995.

[57] T. Roska, L. O. Chua, D. Wolf, T. Kozek, R. Tetzlaff, and F. Puffer, "Simulating nonlinear waves and partial differential equations via CNN - part I: Basic techniques," *IEEE Trans. on Circuits and Systems - I. Fundamental Theory and Applications*, vol. 42, no. 10, pp. 807–815, 1995.

[58] A. N. Evans and M. S. Nixon, "Biased motion adaptive temporal filtering for speckle reduction in echocardiography," *IEEE Trans. on Medical Imaging*, vol. 1, pp. 39–50, 1996.

[59] V. Binzberger, "Preprocessing and calibration methods in real-time 2D and 3D echocardiography (in hungarian)," Master's thesis, Technical University of Budapest, Department of Measurement and Informatics, 2003.

[60] L. Chua and T. Roska, *Cellular neural networks and visual computing.* Cambridge University Press New York, NY, 2002.

[61] A. Zarandy, M. Foldesy, P. Szolgay, S. Tokes, C. Rekeczky, and T. Roska, "Various implementations of topographic, sensory, cellular wave computers," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 5802–5805, 2005.

[62] P. Dudek and P. Hicks, "A general-purpose processor-per-pixel analog SIMD vision chip," *Circuits and Systems I: Regular Papers, IEEE Transactions on [see also Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on]*, vol. 52, no. 1, pp. 13–20, 2005.

[63] C. Rekeczky, I. Szatmari, D. Balya, G. Timar, and A. Zarandy, "Cellular multiadaptive analogic architecture: a computational framework for UAV applications," *Circuits and Systems I: Regular Papers, IEEE Transactions on [see also Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on]*, vol. 51, no. 5, pp. 864–884, 2004.

[64] I. Szatmári, C. Rekeczky, and T. Roska, "A Nonlinear Wave Metric and its CNN Implementation for Object Classification," *The Journal of VLSI Signal Processing*, vol. 23, no. 2, pp. 437 – 447, 1999.

[65] D. Vilarino, D. Cabello, X. Pardo, and V. Brea, "Cellular neural networks and active contours: a tool for image segmentation," *Image and Vision Computing*, vol. 21(2), pp. 189–204, 2003.

[66] D. L. Vilarino and C. Rekeczky, "Pixel-level snakes on the CNN-UM: Algorithm design, on-chip implementation and applications," *International Journal of Circuit Theory and its Applications*, vol. 33, pp. 17–51, 2005.

[67] V. Chalana and Y. Kim, "A methodology for evaluation of boundary detection algorithms onmedical images," *Medical Imaging, IEEE Transactions on*, vol. 16, no. 5, pp. 642 – 652, 1997.

[68] C. Alberola-Lopez, M. Martin-Fernandez, and J. Ruiz-Alzola, "Comments on: A methodology for evaluation of boundary detection algorithms on medical images," *Medical Imaging, IEEE Transactions on*, vol. 23, no. 5, pp. 658 – 660, 2004.

[69] S. Xavier-de-Souza, M. E. Yalcin, J. A. K. Suykens, and J. Vandewalle, "Toward CNN Chip-specific Robustness," *IEEE Trans. Circ. and Systems-I:Fundamental Theory and Appl.*, vol. 51, pp. 892–902, May 2004.

[70] P. Foldesy, A. Zarandy, C. Rekeczky, and T. Roska, "Digital implementation of cellular sensor-computers," *Int. J. Circ. Theor. Appl*, vol. 34, pp. 409–428, 2006.

[71] D Pham et al, "The design and implementation of a first-generation CELL processor," *ISSCC Dig. Tech. Papers*, 2005.

[72] Z. Nagy and P. Szolgay, “Configurable multilayer CNN-UM emulator on FPGA,” *IEEE Trans. on Circ. and Sys. I: Fundamental Theory and Appl.*, vol. 50, no. No. 6., pp. 774–778, 2003.

[73] A. Zarandy and C. Rekeczky, “Bi-i: a standalone ultra high speed cellular vision system,” *Circuits and Systems Magazine, IEEE*, vol. 5, no. 2, pp. 36 – 45, 2005.

[74] L. Vesalainen, J. Poikonen, and A. Paasio, “A fuzzy unit for a mixed-mode array processor,” *Proc. IEEE Workshop on Cellular Nonlinear Networks and its Applications (CNNA)*, 2004.

[75] A. Davis, R. Venkatesan, A. Kaloyeros, M. Beylansky, S. J. Souri, K. Banerjee, K. C. Saraswat, A. Rahman, R. Reif, and J. D. Meindl, “Interconnect limits on gigascale integration (GSI) in the 21st century,” *Proceedings of the IEEE*, vol. 89, pp. 305–324, 2001.

[76] J. D’hooge, B. Bijnens, J. Thoen, F. V. de Werf, G. Sutherland, and P. Suetens, “Echocardiographic strain and strain-rate imaging: a new tool to study regional myocardial function,” *Medical Image Analysis*, vol. 21, no. No. 9., 2002.

[77] A. Stoylen, “Problems and pitfalls in strain rate imaging,” *http://folk.ntnu.no/stoylen/strainrate/Howto/Pitfalls.html and references therein.*

[78] X. Papademetris, A. J. Sinusas, D. P. Dione, R. T. Constable, and J. S. Duncan, “Estimation of 3-D left ventricular deformation from medical images using biomechanical models,” *IEEE Trans. Med. Imag.*, vol. 21, no. 7, 2002.

[79] X. Hao, C. Bruce, C. Pislaru, and J. Greenleaf, “Segmenting high-frequency intracardiac ultrasound images of myocardium into infarcted, ischemic, and normal regions,” *IEEE Trans. Med. Imag.*, vol. 20, no. 12, 2001.

[80] J. W. Hoppin, M. A. Kupinski, G. A. Kastis, E. Clarkson, and H. H. Barrett, "Objective comparison of quantitative imaging modalities without the use of a gold standard," *IEEE Trans. Med. Imag.*, vol. 21, no. 5, pp. 441–449, 2002.

[81] B. Roska and F. Werblin, "Vertical interactions across ten parallel, stacked representations in the mammalian retina.," *Nature*, vol. 410, no. 6828, pp. 583–7, 2001.

[82] G. Rabut and J. Ellenberg, "Automatic real-time three-dimensional cell tracking by fluorescence microscopy," *Journal of Microscopy*, vol. 216, no. 2, pp. 131–137, 2004.

[83] W. Cleveland and S. Devlin, "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *Journal of the American Statistical Association*, vol. 83, no. 403, pp. 596–610, 1988.

[84] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M. Unser, "Design and Validation of a Tool for Neurite Tracing and Analysis in Fluorescence Microscopy Images," *Cytometry Part A*, vol. 58, no. 2, pp. 167–176, 2004.

[85] K. Al-Kofahi, S. Lasek, D. Szarowski, C. Pace, G. Nagy, J. Turner, and B. Roysam, "Rapid automated three-dimensional tracing of neurons from confocalimage stacks," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 6, no. 2, pp. 171–187, 2002.

[86] S. Schmitt, J. Evers, C. Duch, M. Scholz, and K. Obermayer, "New methods for the computer-assisted 3-D reconstruction of neurons from confocal image stacks.," *Neuroimage*, vol. 23, no. 4, pp. 1283–98, 2004.

[87] C. Alonso-Montes, D. Vilarino, and M. Penedo, "CNN-based automatic retinal vascular tree extraction," *Cellular Neural Networks and Their Applications, 2005 9th International Workshop on*, pp. 61–64, 2005.

[88] A. Y. Pogromsky, G. Santoboni, and H. Nijmeijer, "Partial synchronization: from symmetry towards stability," *Physica D*, vol. 172, pp. 65–87, 2002.

[89] I. Belykh, M. Hasler, M. Lauret, and H. Nijmeijer, "Synchronization and graph topology," *Int. Journal of Bifurcation and Chaos*, vol. 15, no. 11, pp. 3423–3433, 2001.

[90] C. W. Wu, "Synchronization in networks of nonlinear dynamical systems coupled via a directed graph," *Nonlinearity*, vol. 18, pp. 1057–1064, 2005.

[91] L. M. Pecora and T. L. Carroll, "Master stability functions for synchronized coupled systems," *Phys. Rev. Lett.*, vol. 80, pp. 2109–2112, Mar 1998.

[92] I. Belykh, V. Belykh, and M. Hasler, "Synchronization in asymmetrically coupled networks with node balance," *Chaos*, vol. 16, no. 1, p. 015102, 2006.

[93] Z. Ma and Z. Liu, "A new method to realize cluster synchronization in connected chaotic networks," *Chaos*, vol. 16, p. 023103, 2006.

[94] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths, "Phase synchronization of chaotic oscillators," *Phys. Rev. Lett.*, vol. 76, pp. 1804–1807, Mar 1996.

[95] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths, "From phase to lag synchronization in coupled chaotic oscillators," *Phys. Rev. Lett.*, vol. 78, pp. 4193–4196, Jun 1997.

[96] H. D. I. Abarbanel, N. F. Rulkov, and M. M. Sushchik, "Generalized synchronization of chaos: The auxiliary system approach," *Phys. Rev. E*, vol. 53, pp. 4528–4535, May 1996.

[97] A. Shabunin, V. Astakhov, and J. Kurths, "Quantitative analysis of chaotic synchronization by means of coherence," *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, vol. 72, no. 1, p. 016218, 2005.

[98] J. A. K. Suykens, P. Curran, and L. Chua, "Robust synthesis for master-slave synchronization of Lur'e systems'," *IEEE Transactions on Circuits and Systems-I*, vol. 46, no. 7, pp. 841–850, 1999.

[99] J. A. K. Suykens and J. Vandewalle, "Generation of n-double scrolls (n=1,2,3,4,...)," *IEEE Transactions on Circuits and Systems-I*, vol. 40, no. 11, pp. 861–867, 1993.

[100] S. Özoguz, A. S. Elwakil, and K. Salama, "n-scroll chaos generator using a nonlinear transconductor," *Electron. Lett.*, vol. 38, no. 14, pp. 685–686, 2002.

[101] J. A. K. Suykens, B. De Moor, and J. Vandewalle, "Static and dynamic stabilizing neural controllers, applicable to transition between equilibrium points," *Neural Networks*, vol. 7, no. 5, pp. 819–831, 1994.

[102] S. Xavier-de-Souza, J. A. K. Suykens, J. Vandewalle, and D. Bollé, "Cooperative behavior in coupled simulated annealing processes with variance control," *Int. Conf. on Nonlinear Theory and its Appl, Bologna (NOLTA)*, 2006.

[103] C. Merkwirth, U. Parlitz, and W. Lauterborn, "Fast nearest-neighbor searching for nonlinear signal processing," *Phys. Rev. E*, vol. 62, pp. 2089–2097, 2000.

[104] B. Coomes, H. Koçak, and K. Palmer, "A Shadowing Theorem for ordinary differential equations," *Zeitschrift für Angewandte Mathematik und Physik (ZAMP)*, vol. 46, no. 1, pp. 85–106, 1995.

[105] F. Rosenblatt, *Principles of Neurodynamics: Perceptions and the Theory of Brain Mechanisms.* Spartan Books, Washington, DC, 1961.

[106] A. Treisman, "Solutions to the binding problem: Process through controversy and convergence," *Neuron*, vol. 24, pp. 105–110, 1999.

[107] P. Milner, "A model for visual shape recognition," *Psychol. Rev.*, vol. 81, pp. 521–535, 1974.

[108] C. von der Malsburg, *The correlation theory of brain function.* Models of Neural Networks II, Springer, 1994. editor E. Domany and J.L. van Hemmen and K. Schulten.

[109] A. Raffone and C. van Leeuwen, "Dynamic synchronization and chaos in an associative neural network with multiple active memories," *Chaos*, vol. 13, no. 3, pp. 1–15, 2003.

[110] C. von der Malsburg, "The what and why of binding: The modeler's perspective," *Neuron*, vol. 24, pp. 95–104, 1999.

[111] I. Belykh, E. de Lange, and M. Hasler, "Synchronization of bursting neurons: What matters in the network topology," *Physical Review Letters*, vol. 94, no. 188101, 2005.

[112] V. N. Belykh, I. Belykh, M. Hasler, and K. Nevidin, "Cluster synchronization in three-dimensional lattices of diffusively coupled oscillators," *Int. Journal of Bifurcation and Chaos*, vol. 13, pp. 755–779, 2003.

[113] R. Sarpeshkar, "Analog-versus-digital extrapolating from electronics to neurobiology," *Neural Computation*, 1998.

[114] D. Bályа, B. Roska, T. Roska, and F. Werblin, "A CNN framework for modeling parallel processing in a mammalian retina," *Int. Journal of Circ. Theory and its Applications*, vol. 30, pp. 363–393, 2002.

[115] C. Allefeld and J. Kurths, "Testing for phase synchronization," *Intl. J. of Bifurcation and Chaos*, vol. 14, pp. 405–416, 2004.

[116] C. Rekeczky, T. Serrano-Gotarredona, T. Roska, and A. Rodriguez-Vazquez, "A stored program 2^nd order/3-layer complex cell CNN-UM," *Proc. Int. Workshop on Cellular Nonlinear Networks and its Applications (CNNA)*, 2000.

[117] *Instant Vision - A computational framework for high speed, multi-core image processing.*

[118] T. Roska, L. Kék, L. Nemes, and Á. Zarándy, "CNN software library: templates and algorithms," tech. rep., Comp. and Auto. Ins. of the Hung. Acad. of Sci, 2005.

[119] The Mathworks Inc., *The Matlab manual.*

Acronyms

3D

Three dimensional. Háromdimenziós. 17, 18, 20, 29, 47, 66, 72, 76

AC

Active contour. Aktív kontúr. 22–24, 29

ACE

Analogic Computing Environment. Analogikai számítási környezet. 49, 55, 60, 61, 63, 64

ADC

Analog to Digital Converter. Analóg-digitális átalakító. 63

ASD

Atrial septal defect. Atrio-szeptális defektus. A pathology, important in pediatric cardiology. In patients with ASD, the wall between the atria contains a hole through which the used blood coming from the veins can mix with the refreshed blood coming from the lung. 17, 18, 22, 41, 50, 51, 58

BW

Black and white. Fekete-fehér. 34, 36, 119

CNN

Cellular Nonlinear Network, a space-time computing paradigm introduced by Leon O. Chua 12, 14, 23, 24, 27, 28, 39, 60–64, 105–107, 115

CNN-UM

Cellular Nonlinear Network Universal Machine, an algorithmically programmable array processor architecture 12, 17

CSA

Coupled Simulated Annealing. Csatolt szimulált lehűtés. 94, 101

CT

Computer Tomography. Számítógépes tomográfia. 66

CWC

Constrained Wave Computing. Kényszerített Hullámszámítás. 17, 23, 29, 30, 34, 35, 50, 55, 61, 62

DAPI

DAPI or 4',6-diamidino-2-phenylindole is a fluorescent stain that binds strongly to DNA. It is used extensively in fluorescence microscopy. DAPI is excited with ultraviolet light. When bound to double-stranded DNA its absorption maximum is at 358 nm and its emission maximum is at 461 nm. This emission is fairly broad, and appears bluecyan. DAPI will also bind to RNA, though it is not as strongly fluorescent. 69–71, 77, 79, 85

DSP

Digital Signal Processor. Digitális jelfeldolgozó processzor. 26, 27, 39, 50, 59–61, 63

FPGA

Field Programmable Gate Array. A field-programmable gate array is a semiconductor device containing programmable logic components called "logic blocks", and programmable interconnects. Logic blocks can be programmed to perform the function of basic logic gates such as AND, and XOR, or more complex combinational functions such as decoders or simple mathematical functions. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memories. 60, 107

fps

Frames per second. Másodpercenként feldolgozott képek száma. 18, 59

GCL

Ganglion Cell Layer. Ganglionsejtes réteg. 70–73, 77–80, 82, 84

GFP

Green Fluorescent Protein. Zölden fluoreszkáló protein. 69, 70, 73, 74, 76, 87

INL

Inner Nuclear Layer. Belső sejtmagos réteg. 70, 72, 73, 77–80, 82

IPL

Belső rostos réteg. A well defined layer in the retina containing the synaptic connections between the ganglion cells and the amacrine or bipolar cells. 13, 69, 71, 73, 80, 87

LOESS

Local regression using weighted linear least squares and a 2^{nd} degree polynomial model. 76, 122

LSM

Laser Scanning Microscope. Lézerrel pásztázó mikroszkóp. 69, 74, 85

LV

Left ventricle. Bal kamra. The left ventricle is the most muscular cavity of the heart pumping oxygenated blood into the main arteries. 17, 19–22, 49, 58, 65

MRI

Magnetic Resonance Imaging. Mágneses rezonancia alapú képalkotás. 20, 21, 60, 66

PC

Personal Computer. Személyi számítógép. 26, 50, 59

PDE

Partial Differential Equation. Parciális differenciálegyenlet. 23, 28

PLS

Pixel Level Snakes, Képelem szintű kígyó 17, 41, 43, 55, 61

RA

Right atrium. Jobb pitvar. 22, 49, 51, 58

RGC

A group cell types that process visual information processed by the outer retina and the amacrine cells and transmit information via the optical nerve toward higher level centres. 69

RT3D

Real-Time 3D. Valósidejű 3D. 20, 21, 60, 66

SNR

Signal to Noise Ratio. Jel-zaj arány. 26, 27

TCAC

Topographic cellular active contour. Topografikus celluláris aktív kontúr. A family of algorithms that solve some image segmentation task using topographic and cellular image processing operators. 18, 24, 27, 29, 47, 49, 59, 60, 67

TEE

Transesophageal echocardiography. Nyelőcsövi echokardiográfia. 51

US

Ultrasound. Ultrahang. 20, 22, 24, 29, 49, 52, 66

VLSI

Very Large Scale Integration. Nagyon nagyfokú integráltság. 105, 106