NONLINEAR WAVE METRIC FOR OBJECT COMPARISON ON CNN ARCHITECTURE

Ph.D. Dissertation

István Szatmári

Scientific adviser:

Tamás Roska, D.Sc.

Analogical and Neural Computing Laboratory
Computer and Automation Institute
Hungarian Academy of Sciences

Budapest, 2001

"Everything should be made as simple as possible, but not simpler."

Albert Einstein

Acknowledgements

First and foremost I would like to thank Prof. Tamás Roska for his conscientious and faithful support and encouragement. He gave me continuos attendance through the years that I could spent in the Analogical and Neural Computing Laboratory. His unbroken enthusiasm, commitment for a thorough work, and also his personal qualities made an example for me and helped so much in my work.

I owe thanks to Prof. Leon O. Chua who invited me and I could spend a semester in his Nonlinear Electronic Laboratory at Berkeley working on different problems.

I would like to thank my colleagues in the Analogical Laboratory whose work made it possible to carry out my experiments efficiently. I thank Péter Szolgay, András Radványi, and Tamás Szirányi for their help in my first research period. I worked much together with Ákos Zarándy, László Kék, and Károly László in different projects and I learned not only professional skills but also won good fellowship. Especially I am very grateful to Csaba Rekeczky with whom we talked several times about my research problems and I could explore so many possibilities.

I had some great teachers who should be mentioned here. Prof. Pál Rózsa, György Fodor, Zoltán Füzessy, Endre Selényi, Gábor Péceli; Rudolf Nehéz, Jánosné Orosz, János Horváth, and István Kundrák.

I owe thanks to my friends who gave me other type of help that is necessary to a complete and full life.

I am very grateful to my parents who always believed me and in my goals.

And last but not least I thank my wife Eve for being always so patient and supporting friend to whom I dedicate this dissertation.

Contents

Notation and Definitions

Cellular Neural/Nonlinear Networks

u_{ij}, x_{ij}, y_{ij}	-	input, state and output of a specified CNN cell
ij	-	grid (or node) point identifier
$kl \in S_r$	-	grid point identifier in the neighborhood of ij within the radius r
$\Delta v, \Delta_1 v, \Delta_2 v$		are the differences of any two cell variables within S_r of two layers, of the kl-th and ij-th node
$A_{ij,kl}$	-	linear feedback term
$B_{ij,kl}$	-	linear control term
$C_{ij,kl}$	-	linear state-feedback term
$\hat{A}_{ij,kl}$	-	nonlinear feedback term
$\hat{B}_{ij,kl}$	-	nonlinear control term
$\hat{C}_{ij,kl}$	-	nonlinear state-feedback term
$\hat{D}_{ij,kl}$	-	generalized nonlinear term
$f(.)$	-	a sigmoid-type function, the output characteristics of a CNN cell
R_{ij} *or* R	-	linear resistor of a CNN cell
C_{ij} *or* C	-	linear capacitor of a CNN cell
τ	-	time constant of a CNN cell
z_{ij}	-	(space-variant) cell current (also called as bias or threshold) of a CNN cell

Nonlinear wave metric (Chapter 3)

$S(A,B)$	-	similarity measure between A and B objects or feature vectors
$D(A,B)$ *or* $d(A,B)$	-	distance or distance function between A and B objects or feature vectors
L_1, L_2, L_∞	-	Manhattan (city-block), Euclidean, Chebychev norms, respectively
$HD(A,B)$	-	Hamming distance between A and B objects or feature vectors
$HsD(A,B)$	-	Hausdorff distance between A and B objects or feature vectors
$HsD(u)$	-	local Hausdorff distance value at spatial position $u(x,y)$
$WD(A,B)$	-	Weighted Hamming (Integrated Hausdorff) distance between A and B objects
$Q_D(h)$		sensitivity or sensitivity function of a given distance measure D to a given perturbation h (position, noise, scale, …)
U_A	-	binary image of object A
V_A	-	gary-scale image of object A
$W_{AB}(.)$	-	spatio-temporal wave mapping between objects A and B
$w_{AB}(.)$	-	contribution function of the wave mapping between objects A and B
$\Psi_{AB}(.)$	-	time distribution of the wave mapping between objects A and B
$\Phi_{AB}(.)$	-	spatial distribution of the wave mapping between objects A and B

$G_{AB}(.)$	-	gray-scale morphology operation between objects A and B
$B_{AB}(.)$	-	binary morphology operation between objects A and B

Spatio-temporal dynamics in classification algorithms (Chapter 4)

$\Phi(t)$	a two dimensional image flow
$\Psi(\Phi)$	a function or a functional on an image flow
α *or* α_{pq}	a spatio-temporal operation between p-th and q-th layers
MAT	*Medial Axis Transform*, gray-scale intensity map at skeleton of object
DT	*Distance Transform*, distance value computation of object's pixels to background
DM	*Distance Map*, gray-scale intensity map of DT
CP	*Center Points* containing local center points of objects
RM	*Radii Map*, gray-scale intensity map containing size related information of objects
ROC	*Response Operating Characteristic*, diagram showing true positive values versus false positive values

1 INTRODUCTION

Comparing two objects, usually a reference or a model to an unknown object, is fundamental task in many image processing and classification type applications. These systems make decisions and these decisions are based on some distance calculations. But what type of distance operation should be used? If the choice is optimal then comparison will separate objects correctly. Otherwise sophisticated investigations are necessary to extract different features of objects in order to find good characteristic properties. Here also comes to fore distance operations among features. Of course there is not general answer and there is not "best method" for all types of comparisons.

In an engineering problem researchers usually choose already well-tested methods and try to adapt them for their applications. But they often forget to investigate the problem of distance calculation. My aim was to address the basics of comparison, the distance calculation itself. The focus will be put on image processing related problems but results, in my belief, can also be used in other fields. The main contribution of this work is that it highlights the potentiality of *spatio-temporal dynamical processes* in object shape comparison and distance calculation. Using nonlinear dynamical processes (e.g. propagating waves) for exploring geometrical properties of objects opens a novel approach for analysis. Along spatial coordinates as a new dimension, time related information can be extracted and makes possible to investigate "hidden" properties of objects. These characteristics enable to solve also untreatable problems.

Use of spatio-temporal dynamical processes in a real time object comparison and classification is a good example, where the demand for a supercomputing power makes digital and sequential design inappropriate in the near future. Massively parallel architectures are necessary to provide medium where these nonlinear dynamical processes can take place. Computing architectures based on the Cellular Neural/Nonlinear Network (CNN) paradigm [1-5] and CNN Universal machine (CNN-UM) [4] offer an adequate solution for the stated problem. 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 fixed or non-fixed point states of the network. The spread and utilization of these novel computing tools in industrial applications requires additional elements: the possibility of quick reprogramming and temporary storage capability. Completing the base cells of CNN with local sensors, data memories, arithmetical and logical units, furthermore with global program memories and control units results in the CNN Universal Machine (CNN-UM) architecture [4]. The CNN-UM is an analogic (analog and logic) supercomputer, universal both in Turing sense and as a nonlinear operator [6], therefore it can be used as a general architectural framework in designing CNN processors [11], [12], [13].

In this dissertation, the focus is put on designing comparison metrics based on spatio-temporal nonlinear wave propagation in a dynamical medium. The approach taken is mainly motivated by applications thus the majority of algorithms rely on the CNN architecture and templates used in algorithms are such that their prototypes can be tested on existing VLSI implemented CNN chips. Notwithstanding, the theoretical considerations are general and can be discussed without any concrete physical implementation.

The second major part of this work exploits the previous results and demonstrates the potentiality of spatio-temporal dynamical processes not only for comparison but in a more general way where they play the major role in model based object classification algorithms. What makes necessary that we are looking for novel type of problem handling in classification algorithms? We are facing a revolutionary process in electronic and computer industry. This is the revolution of the sensor technology and in these new devices thousands of analog signals are produced continuously in time and are waiting for processing. The statement is that a new computing paradigm is needed [7]. The signals of these sensors are mainly arranged into a 2D array and this flow is to be processed in real time. Operations on this flow are concerned as spatio-temporal instructions and they may be in many others processes based on nonlinear wave phenomena and different nonlinear Partial Differential Equations (PDEs). The exciting challenge is how to combine them into useful algorithms.

What I try to demonstrate in this part of my work is that spatio-temporal dynamical processes can be used as elementary operations in algorithms and these elementary analogical operations can solve very difficult classification problems. The motivation is that in an efficient computational framework such real time applications can be tackled where traditional sequential design would not be feasible.

The dissertation is organized as follows. Chapter 2 describes the CNN frame of computing including the CNN Universal Machine architecture and gives some information about existing hardware architectures. Chapter 3 (*Thesis I*) presents the development of the class of nonlinear wave metric for 2D object shape comparison. The novelty of this approach relies on the use of spatio-temporal dynamical processes for object investigation. In Chapter 4 (*Thesis II*) it is discussed how spatio-temporal dynamics as elementary operators can be used in algorithms designed for model based object classification. It exploits the theoretical result of Chapter 3 and gives a comprehensive study of a specific image processing application: bubble-debris separation experiment in the task of condition based maintenance of engines. Chapter 5 summarizes the main results and highlights some potential application areas where contribution of this thesis could be efficiently used.

2 THE CNN FRAME OF COMPUTING

In this chapter the CNN architecture is defined from the cell level up to the general algorithmic level. First, the CNN paradigm is described along with the mathematical description of the core cell and the cell interactions. Then the CNN Universal Machine is introduced. This stored program array processor hosts as the general hardware platform for all experiments.

2.1 The CNN Paradigm

A Cellular Neural/Nonlinear Network (CNN) is defined by the following principles [1-3]:

- A spatially discrete collection of continuous nonlinear dynamical systems called cells where information can be encrypted into each cell via three independent variables called input, threshold, and initial state.
- A coupling law relating one or more relevant variables of each cell to all neighboring cells located within a prescribed sphere of influence $S_r(ij)$ of radius r centered at ij.

Fig. 2.1 shows a 2D rectangular CNN composed of cells that are connected to their nearest neighbors. Due to its symmetry, regular structure and simplicity this type of arrangement (a rectangular grid) is primarily considered in all implementations.

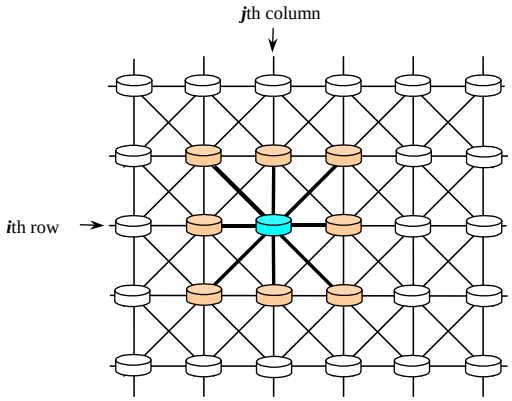


Figure 2.1 A 2-dimensional CNN defined on a square grid. The *ij*-th cell of the array is colored by light blue, cells that fall within the sphere of influence of neighborhood radius $r = 1$ (the nearest neighbors) by light brown.

2.2 CNN Core Cell and the Inter-cell Interactions

The CNN paradigm does not specify the properties of a cell. As the basic framework throughout this dissertation, let us consider a two dimensional (*MxN*) CNN array in which the cell dynamics is described by the following nonlinear ordinary differential equation with linear and nonlinear terms (the extension to three dimensions is straightforward allowing similar interlayer interactions). In this work three different cell models were used as follows.

- *First order cell model*

This is the standard first order CNN cell. This model is used in the CNN-UM chip made in Berkeley [10].

- *Second order cell model*

This is a second order CNN cell, which is the same as the previous except for an additional capacitance connected across the output of the cell.

- *Full range first order cell model*

A first order cell, the state and output are the same and the voltage swing of the state transient is limited within [-1;1]. This model is used in the CNN-UM chip made in Seville [11, 12].

State equation of a single layer CNN with first order cell model

The standard first order CNN array dynamics is described by the following equations.

$$
\begin{aligned}
C\dot{x}_{ij}(t) = & -\frac{1}{R}x_{ij}(t) + z_{ij} && \text{cell dynamics} \\
& \sum_{kl \in S_r(ij)} A_{ij;kl} \cdot y_{kl}(t) + \sum_{kl \in S_r(ij)} B_{ij;kl} \cdot u_{kl}(t) + && \text{linear cell} \\
& \sum_{kl \in S_r(ij)} C_{ij;kl} \cdot x_{kl}(t) + \sum_{kl \in S_r(ij)} D_{ij;kl}(\Delta v) + && \text{interactions} \\
& \sum_{kl \in S_r(ij)} \hat{A}\left(y_{ij}(t), y_{kl}(t)\right) + \sum_{kl \in S_r(ij)} \hat{B}\left(u_{ij}(t), u_{kl}(t)\right) + && \text{nonlinear cell} \\
& \sum_{kl \in S_r(ij)} \hat{C}\left(x_{ij}(t), x_{kl}(t)\right) + \sum_{kl \in S_r(ij)} \hat{D}\left(\Delta_1 v\right) \cdot \Delta_2 v && \text{interactions}
\end{aligned}
\tag{1}
$$

$$
y_{ij}(t) = f(x_{ij}(t)) = \begin{cases} 1 & \text{if } x_{ij}(t) \geq 1 \\ x_{ij}(t) & \text{if } -1 \leq x_{ij}(t) \leq 1 \\ -1 & \text{if } x_{ij}(t) \leq -1 \end{cases} \quad \text{output equation} \tag{2}
$$

where

- x_{ij}, y_{ij}, u_{ij} are the state, the output, and the input voltage of the specified CNN cell, respectively, The state and output vary in time, the input is static (time independent), *ij* refers to a grid point associated with a cell on the 2D grid, and $kl \in S_r$ is a grid point in the neighborhood within the radius *r* of the cell *ij*.
- z_{ij} is the cell current (also referred to as bias or threshold) which could be space and time variant.
- Term $A_{ij,kl}$ represents the linear feedback, $B_{ij,kl}$ the linear control, and $C_{ij;kl}$ the linear feedback from state. $D_{ij,kl}$ is a difference controlled linear template.
- $\hat{A},\hat{B},\hat{C}$ are the nonlinear templates, $\hat{D}$ is a difference controlled nonlinear template.
- Term *C* and *R* are the capacitance and resistance of the cells, respectively, default values are *C=1*, *R=1*.
- The Δv, $\Delta_1 v$, and $\Delta_2 v$ are the differences (or other arithmetic operations) of any two cell variables within S_r of two layers,
- Term *f(.)* is the output nonlinearity, in our case a unity gain sigmoid.
- The *t* is the continuous time variable.

The first part of Equation (1) is called cell dynamics, the following additive terms represent the synaptic linear and nonlinear interactions. Though the threshold z_{ij} may be space-variant, usually it is added to the template (space-invariant case). Equation (2) is the output equation.

A CNN base cell corresponding to is shown in Fig. 2.2.

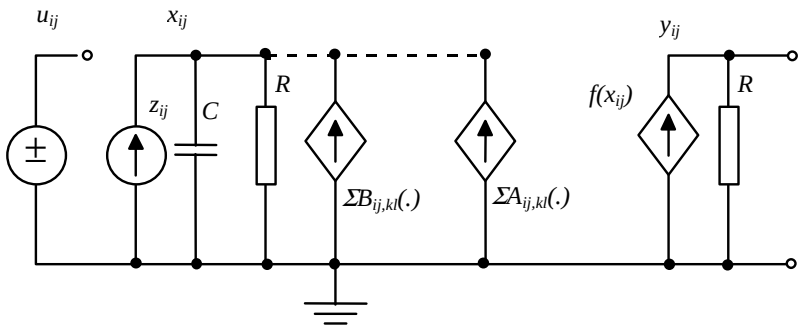


Figure 2.2 A CNN base cell corresponding to the equations (1), (2). The linear control and feedback terms are represented by voltage controlled current sources ($B_{ij,kl}$ and $A_{ij,kl}$).

The time constant of a CNN cell is determined by the linear capacitor (*C*) and the linear resistor (*R*) and it can be expressed as $\tau=RC$. A CNN cloning template, the program of the CNN array, is given with the linear and nonlinear terms completed by the cell current.

Equations (1), (2) define a rather complex framework for computation. This is used only in nonlinear template design to make it clear what exactly the synaptic interactions are solving specific problems.

State equation of a multi-layer CNN with first order cell model

In a multi-layer CNN of L layers, the *p*th layer dynamics is described by:

$$C_p \dot{x}_{p,ij}(t) = -\frac{1}{R_p} x_{p,ij}(t) + \sum_{q=1}^{L} \ldots \left\{ T1_p^q(ij;kl) \cdot y_{q;kl} + T2_p^q(ij;kl) \cdot u_{q;kl} + z_{p;ij} \right\} \ldots \qquad (3)$$

where

- L is the number of layers of the network,
- T_q^p -s are linear or nonlinear templates,
- q : *from* layer. It specifies the layer from the interactions are computed.
- p : *on* layer. It determines the actual layer on which the template values and current are.

The last part denotes the multi-layer synaptic interactions. For sake of simplicity, not all individual template interactions are detailed. In some cases (e.g. biological modeling) it is useful to call the terms T_q^p the signal transfer terms since whether they represent a feedback or a control (feed-forward) depends on the whole network model. The default values are $C_p = 1$, $R_p = 1$.

State equation of a single layer CNN with second order cell model

If the cell is a second order type, the state equation is the same as equation (1), except the output equation has the following form (single layer case):

$$C_y \dot{y}_{ij}(t) = -\frac{1}{R_y} y_{ij} + f(x_{ij}(t)),$$
$$f(x_{ij}(t)) = \begin{cases} 1 & \text{if } x_{ij}(t) \geq 1 \\ x_{ij}(t) & \text{if } -1 \leq x_{ij}(t) \leq 1 \\ -1 & \text{if } x_{ij}(t) \leq -1 \end{cases} \qquad (4)$$

where

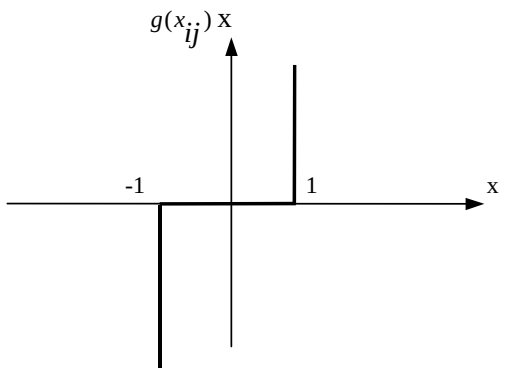
- C_y is the capacitance connected across the output while R_y is the output resistance. The default values are $C_y = 1, R_y = 1$.

State equation of a single layer CNN with full range first order cell model

In case of full range cells the state equation is the same as in case of first order cell except that the cell dynamics and output equation are:

$$\dot{x}_{ij}(t) = -g(x_{ij}(t)) + z_{ij} \quad \text{cell dynamics}$$
$$y_{ij}(t) = x_{ij}(t) \quad \text{output equation} \qquad (5)$$

where $g(x_{ij}(t))$ is defined as



2.3 The CNN Universal Machine and CNN Universal Chips

All early neural network chip realizations had a common problem: they implemented a single instruction only, thus the weight matrix was fixed when processing some input. Reprogramming (i.e. changing the weight matrix) was possible for some devices but took in order of magnitudes longer time than the computation itself.

This observation motivated the design of the CNN Universal Machine (CNN-UM, [4]), a stored program nonlinear array computer. This new architecture is able to combine analog array operations with local logic efficiently. Since the reprogramming time is approximately equal to the settling time of a non-propagating analog operation it is capable of executing complex analogic algorithms. To ensure programmability, a global programming unit was added to the array, and to make it possible an efficient reuse of intermediate results, each computing cell was extended by local memories. In addition to local storage, every cell might be equipped with local sensors and additional circuitry to perform cell-wise analog and logical operations. The architecture of the CNN-UM is shown in Fig. 2.3.

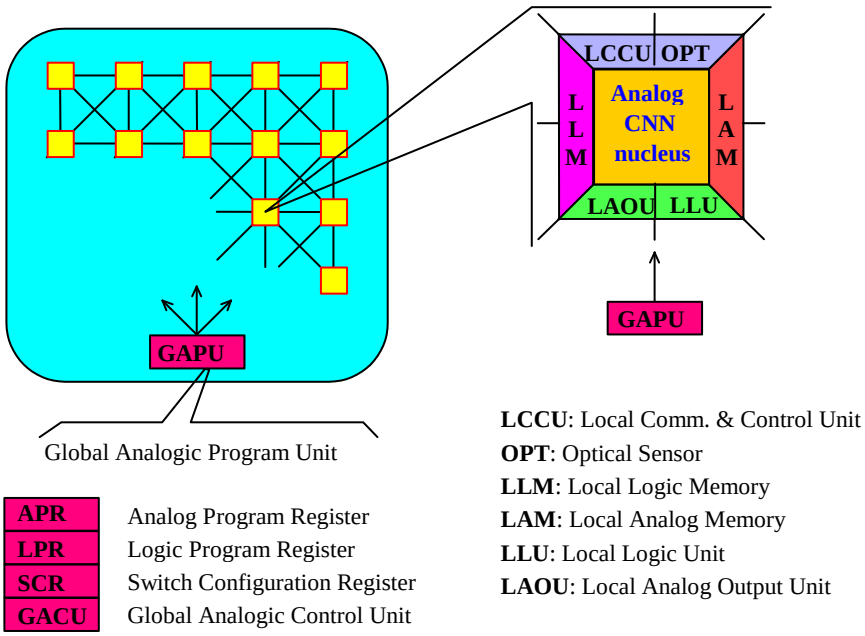


Figure 2.3 The architecture of the CNN Universal Machine, the analogic array supercomputer.

As illustrated in Fig. 2.3 the CNN-UM is built around the dynamic computing core of a simple CNN. An image can be acquired through the sensory input (e.g. OPT: Optical Sensor). Local memories store analog (LAM: Local Analog Memory) and logic (LLM: Local Logical Memory) values in each cell. A Local Analog Output Unit (LAOU) and a Local Logic Unit (LLU) perform cell-wise analog and logic operations on the stored values. The output is always transferred to one of the local memories. The Local Communication and Control Unit (LCCU) provides for communication between the extended cell and the central programming unit of the machine, the Global Analogic Programming Unit (GAPU). The GAPU has four functional blocks. The Analog Program Register (APR) stores the analog program instructions, the CNN

templates. In case of linear templates, for a connectivity $r = 1$ a set of 19 real numbers have to be stored (this is even less for both linear and nonlinear templates assuming spatial symmetry and isotropy). All other units within the GAPU are logic registers containing the control codes for operating the cell array. The Local Program Register (LPR) contains control sequences for the individual cell's LLU, the Switch Configuration Register (SCR) stores the codes to initiate the different switch configurations when accessing the different functional units (e.g. whether to run a linear or nonlinear template). The Global Analogic Control Unit (GACU) stores the instruction sequence of the main (analogic) program. The GACU also controls the timing, sequence of instructions and data transfers on the chip and synchronizes the communication with any external controlling device.

Synthesizing an analogic algorithm running on the CNN-UM the designer should decompose the solution in a sequence of analog and logical operations. A limited number of intermediate results can be locally stored and combined. Some of these outputs can be used as a bias map (space variant current) or fixed-state map (space-variant mask) in the next operation adding spatial adaptivity to the algorithms without introducing complicated inter-cell couplings. Analog operations are defined by either linear or nonlinear templates. The output can be defined both in fixed and non-fixed state of the network (equilibrium and non-equilibrium computing) depending on the control of the transient length. It can be assumed that elementary logical (**NOT**, **AND**, **OR,** *etc.*) and arithmetical (**ADD**, **SUB**) operations are implemented and can be used on the cell level between LLM and LAM locations, respectively. In addition data transfer and conversion can be performed between LAMs and LLMs.

3 NONLINEAR WAVE METRIC FOR 2D OBJECT COMPARISON

In this chapter a nonlinear wave metric is introduced and discussed for object shape comparison and object classification. The goal was to develop a shape comparison method that is efficient to compute, and produces intuitively reasonable results. It will be shown that the choice of a metric is a nontrivial problem since it is easy to give examples when well-known distance measures, such as Hamming, Hausdorff, and Nonlinear Hausdorff metrics are completely inadequate for this classification. As an alternative a generalized theorem is proposed that includes the previous metrics as special cases, and what is more, it overcomes their disadvantages. It is based on nonlinear wave propagation and defines a computational framework that is well-suited for parallel array processors.

First, some notes will be given about metrics, distances, and related topics. Without completeness about this subject it might give some inspirations for further work.

Second, the implementation of the autowave distance on the CNN architecture will be introduced. Autowave distance is a nonlinear variant of the Hausdorff metric. It will be shown that the proposed framework is especially efficient to calculate this type of metric. On the other hand, to estimate the Hausdorff and the autowave metric on sequential type of hardware requires time consuming and sophisticated algorithms.

Third, limitations of commonly used metrics such as Hamming and Hausdorff distances will be discussed and shown how they might fail to measure differences among objects. To overcome this problem, these two distance measures can be combined by using dynamic processes. The investigated weighted Hamming (integrated Hausdorff) metric has several advantages over its ascendants.

Finally, a generalized theorem is introduced for object comparison based on spatio-temporal dynamical processes. This process acts as a transformation stage before comparison in which waves propagating on objects explore shape properties of objects and the extracted information serves as a common database for further analysis. A few examples will be given what kind of measures can be determined.

Different CNN architectures and solutions are investigated for the proposed wave metric and analyzed its VLSI implementation complexity. The single-layer trigger wave generation and iterative metric computation can be tested on present CNN-UM chips. The two-layer implementation of wave type metric results in a flexible and efficient tool for object classification. The parallel computing capability is emphasized and the limitations are also thoroughly investigated.

3.1 Preliminaries

Pattern recognition and object classifications are central problems in image processing. Their major objective is to determine the extent to which one shape differs from another. There are several methods that can all be viewed as techniques for image classification or recognition via comparison with prototypes (pattern matching). This comparison requires the measurement of point sets and should be able to determine the difference and/or the level of the similarity. This comparison measurement is usually based on a distance computation. Distance functions are used in many approaches as basic tool. The efficiency of an application depends strongly on the chosen distance measure. On the other hand, the type of a distance computation influences what kind of distance measures might be optimal and the choice is not trivial. In order to show the complexity of this problem, we have to overview the field where distance measures might play roles. This overview focuses not only to the field of image processing, but more generally, tries to show a landscape about this topic also including, for instance, cognitive psychology. This might help to understand typical problems and to show possible directions for solutions.

First, we should clarify two terms and highlight that although they describe similar concepts, they are different.

- *Matching* is the fundamental operation in many fields (e.g. object classification, query in databases, [33]), and it consists of comparing an object (*item*) with a model (*query*), and deciding whether the item satisfies the query or not. Using robust matching techniques result in often not satisfactory solution, they are still based on the idea of matching, and it states basically that the object and the model ought to be equal; they just happen to be slightly different due to some accident or imperfection.

- *Similarity measure,* rather, orders the objects with respect to similarity with the model, given a fixed similarity criterion. In practice, of course, we are interested only in a tiny fraction of the objects, the objects most similar to the model. Since it is the human user that, in the end, has to be satisfied with the results of the query, it is natural to base the similarity measure that we will use on the characteristics of human similarity assessment.

In case of database search, for instance, the main difference between match-based and similarity searches can be stated as follows. The result of a match-based search is a partition of the database in the set of items (for instance images) that match the query, and the set of items that do not. The result of a similarity search is a permutation (in particular, a sorting with respect to the similarity criterion), of the whole database. Matching techniques are developed mostly for recognition of objects under several conditions of distortion. Similarity measures, on the other hand, are used in applications, like image databases, in which the query image is just a very partial model of the user's desires to find similar images. The requirement of this application is that a similarity measure should accurately predict perceptual similarity for all, for instance, images reasonably similar to the query. Matching and similarity measurements are not seldom based on the same techniques, but they differ in emphasis and applications.

Measuring meaningful image similarity is a twofold problem that rests on two elements: finding a set of features which adequately encodes the characteristics that we intend to measure, and endowing the feature space with a suitable metric. Since the same feature space can be endowed with an infinity of metrics, the two problems are by no means equivalent, nor does the first subsume the second. Typically, the metric space is assumed to be Euclidean, although many

researchers made models that challenge the Euclidean distance assumption in non trivial ways. Without going into many details, we have to analyze alternatives to this assumption.

3.1.1 Models of Similarity

A similarity computation is defined by the relationships between two objects under a chosen similarity model. In theory, the objects in question could be anything; for example, geometric patterns, the condition of a production rule, semantic feature lists or a signal of some sort. Human similarity perception has been extensively studying by psychologists for many years and these results can be useful for computer scientists. It is impossible to give more than a brief sketch, interested readers are advised to consult the cited literature for a more comprehensive discussion [56], [57], [58]. Two basic models have been used extensively in cognitive psychology; the *geometric model* and the *contrast model*.

3.1.1.1 The Geometric Model of Similarity

In case of geometric models, they hypothesized that similarity assessment was based on the measurement of a suitable distance in a psychological space [45], [47], [49]. The features of objects were transformed into points of this space and the distance between two feature-sets determines the similarity of objects [53]. This point of view implies that the distance function on which similarity is based must satisfy the metric axioms. These axioms are

- constancy of self-similarity,
- minimality,
- symmetry,
- triangle inequality.

The more formal definition is given later. The constancy states that any object to itself has the same similarity level. The minimality describes an object being most similar to itself. All other objects naturally cannot be more similar. Symmetry holds that A is as similar to B as B is to A. The triangle inequality states that the differences between any three objects are greater or equal to the differences between two of those three. Measures of similarity defined by this method are evidently context-free, however, empirical findings has been that similarity judgements are highly context-sensitive [50]. Theories differ in the way they deal with the properties of geometric distance, and by the number and nature of distance axioms they accept or refuse. For instance, the constancy of self-similarity has been refuted by Krumhansl in [55]. The second and third axioms, namely, the minimality and symmetry are also open to experimental investigation. Tversky in [51] argued that these assumptions may sometimes inappropriate and showed cases in which the symmetry does not hold. Epistemologically, the triangle inequality is the weakest axiom. Generally acknowledged that at least for some types of stimuli the triangle inequality does not hold [46], [52].

From these notes, it seems like the geometric model cannot really be used because all the four basic axioms are questionable. In spite of these problems, metric models are widely used also in psychology, with some adjustments to account for the failure of the distance axioms. In the following , we review briefly some of these models.

In a very influential paper [45], the Euclidean nature of perception was debated and shown examples where instead of using Euclidean distance the city-block distance predicted right

observation measures. Krumhansl [55] proposed a modification to the geometric distance model to account for some violations of the distance axioms. Her idea is known as quasi-distance model. An important class of metric models was introduced by [53], [54], [49] which is referred as Thurstone-Shepard similarity model. This model is based on identification and generalization data, that is, on the probabilities that the responses learned for a given stimulus. An approach based on similar premises was pursued by [46] and belongs also to stochastic models.

3.1.1.2 The Contrast Model of Similarity

The contrast model has been used to account for several empirical phenomena where the geometric models fail. In [51], Tversky proposed his famous feature contrast model. Instead of considering stimuli as points in a metric space and abandoning the metric axioms [52], Tversky characterized stimuli as sets of features. If A and B are the features of objects or stimulus, then the general family of Tversky similarity measures can be written as

$$S(A,B) = f(A \cap B) - \alpha f(A - B) - \beta f(B - A) \tag{6}$$

i.e. expressing similarity as a linear combination of the measures of the common (An B) and the distinctive (A-B, B-A) features. He introduced two further axioms of the theory, namely, the monotonicity and the independence. It has been shown that the contrast model both includes and improves upon the geometric model and can account for violation of all the geometric distance axioms. However, there are studies on which these model fails [58]. Problems belong both theoretical and empirical cases. One of the theoretical difficulties facing the contrast model is the problem of defining the features of objects in a priori fashion. The empirical difficulties for the contrast model arise from the effects of presentation ordering and the effects of prior history on similarity judgements. Researchers to overcome these problems tried different possibilities, one of them introduced a dynamic similarity model where these effects are treated efficiently [58].

As we have seen, the problem of classification based on matching techniques, difference computation or similarity judgement is a difficult topic and there is a lack of a general theorem. Many studies were carried out on this subject addressing also basic questions, for instance, metric axioms, and shown both theoretic and empirical difficulties. As a consequence, endowing an image processing task with a suitable metric requires thorough investigation. Next we will review some distance functions and their properties used in different applications especially focusing attention on image processing.

3.2 Metrics and their Properties

Distance functions are used in many fields including pattern recognition [59],[60],[61], computational geometry [62], cluster analysis [63], neural networks [48], [64], [65], [66], instance-based learning [67], [68]. [69], [70], machine learning [71], [72], statistics [73], cognitive psychology (see previous mentioned references), and philosophy of science [74], [75], [76], [77]. Several of these functions are shown in Figure 3.1. Comprehensive discussion of distance functions can be found in [78], [79].

Minkowsky : **Euclidean (L_2) :** **Manhattan / city - block (L_1) :**

$$d(A,B)=\left(\sum_{i=1}^{m}\left|A_i-B_i\right|^{r}\right)^{\frac{1}{r}} \qquad d(A,B)=\sqrt{\sum_{i=1}^{m}(A_i-B_i)^2} \qquad d(A,B)=\sum_{i=1}^{m}\left|A_i-B_i\right|$$

Camberra : $d(A,B)=\sum_{i=1}^{m}\frac{\left|A_i-B_i\right|}{\left|A_i+B_i\right|}$ **Chebychev / chessboard (L_∞) :** $d(A,B)=\max_{i=1}^{m}\left|A_i-B_i\right|$

Quadratic : $d(A,B)=(A-B)^T\boldsymbol{Q}(A-B)=\sum_{j=1}^{m}\left(\sum_{i=1}^{m}(A_i-B_i)Q_{ji}\right)(A_j-B_j)$

$\boldsymbol{Q}$ is a problem specific positive definit *m x m* weight matrix.

Mahalonobis : $d(A,B)=\left[\det\boldsymbol{V}\right]^{\frac{1}{m}}(A-B)^T\boldsymbol{V}^{-1}(A-B)$

$\boldsymbol{V}$ is the covariance matrix of $C_1 \ldots C_m$, and C_j is the vector of values for attribute j occuring in the training set instances 1...n.

Correlation : $$d(A,B)=\frac{\sum_{i=1}^{m}(A_i-\overline{A_i})(B_i-\overline{B_i})}{\sqrt{\sum_{i=1}^{m}(A_i-\overline{A_i})^2\sum_{i=1}^{m}(B_i-\overline{B_i})^2}}$$

$\overline{A_i}=\overline{B_i}$ and is the average value for attribute *i* occuring in the training set.

Chi - square : $d(A,B)=\sum_{i=1}^{m}\frac{1}{sum_i}\left(\frac{A_i}{size_A}-\frac{B_i}{size_B}\right)^2$

sum_i is the sum of all values for attribute *i* occuring in the training set,
and $size_A$ and $size_B$ are the sums of all values in the vector A and vector B, respectively.

Kendall's Rank Correlation : $d(A,B)=1-\frac{2}{n(n-1)}\sum_{i=1}^{m}\sum_{j=1}^{m}sign(A_i-A_j)sign(B_i-B_j)$

$sign(x)=-1,\ 0,$ or 1 if $x<0,\ x=0,$ or $x>0,$ respectively.

Figure 3.1 Equations of some distance functions (A and B are vectors of m attribute values).

Although there have been many distance functions proposed, by far the most commonly used is the Euclidean distance function, which is defined as

$$d_E(A,B)=\sqrt{\sum_{i=1}^{m}(A_i-B_i)^2} \tag{7}$$

where A and B are two input vectors (one typically is a stored instance of a model, and the other an input vector to be classified) and m is the number of input variables (features) in the application. The square root is often not computed in practice, because the closest objects will still be the closest, regardless of whether the square root is taken.

An alternative function, the city-block or Manhattan distance function, requires less computation and defined as

$$d_{MC}(A,B)=\sum_{i=1}^{m}\left|A_i-B_i\right| \tag{8}$$

The Euclidean and Manhattan distance functions are equivalent to the Minkowsky distance function with r = 2 and 1, respectively.

One weakness of the distance functions, mentioned above, is that if one of the input attributes has a relatively large range, then it can overpower the other attributes. Therefore, distances are often normalized by dividing the distance for each attribute by the range of that attribute, or it is also common to divide by the standard deviation, or using other weighting schemes.

As we have seen, object comparison rests on two elements: finding a set of features to encode object's properties, and endowing the feature space with a suitable metric. A metric should use some kind of distance function and it seems that there is no distance function that can be strictly better than any other in terms of generalization ability, unless to a specific kinds of problems.

Next, we will focus on image processing applications where objects are parts of images and our aim is to state assessments about metrics in this framework. In the geometric approach the comparison task can be defined as a distance measurement between two objects, see Figure 3.2. The feature extraction might be incorporated into the distance computation as we will have seen.

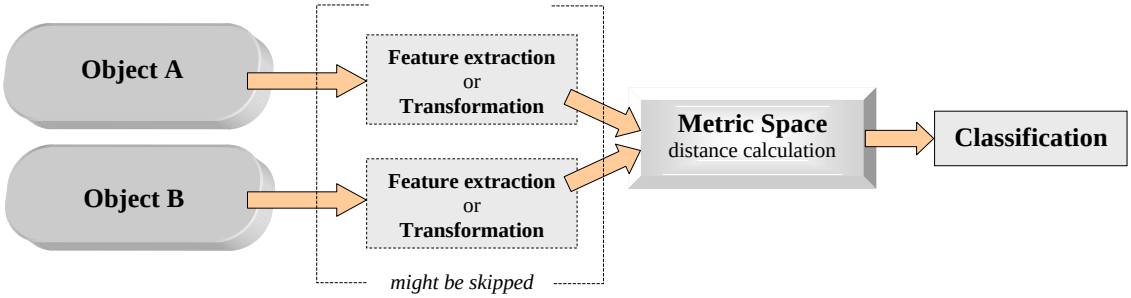


Figure 3.2 General scheme of classification process using distance calculation. Distance computation is based on a metric space. The feature extraction is not absolutely necessary or it can be involved into the distance calculation.

A function $D : S \times S \rightarrow \Re^{+}$ is a *metric* on a nonempty set *S*, if for all $A, B, C \in S$ we have

$$
\begin{aligned}
&1.\, D_{AB} = D(A,B) \geq 0 \text{ and } \mathrm{D}_{AB} = 0 \text{ if and only if } A = B. \text{ (positiveness)} \\
&2.\, \mathrm{D}_{AB} = D_{BA} \text{ (symmetry)} \\
&3.\, \mathrm{D}_{AB} + D_{BC} \geq D_{AC} \text{ (triangle inequality)}
\end{aligned}
\tag{9}
$$

A measure is defined as distance if it fulfils the first two postulates and it is metric if in addition the third, triangle inequality also holds. These conditions are very similar to the similarity measure where constancy and minimality stand for positiveness. It states that any object to itself is closest and all other objects naturally cannot be closer. Symmetry holds that *A* is as far to *B* as *B* is to *A*. The triangle inequality states that the differences between any three objects are smaller or equal to the differences between two of those three. In certain contexts, it is natural or desired that a measure of distance between point sets is a metric, i.e. it satisfies the postulates of a distance function and in addition the triangle inequality. Our aim is to keep these conditions for developing comparison technique unless we need a less strict method.

Several measures can be defined as distance functions. Given a distance many questions may arise. Does it describe the difference between objects well? Do we have any information about similarity, shape, geometry (structure, morphology), etc? Can this distance function be used for classification? We will discuss two well-known distances used in image processing applications

focusing on binary images containing objects to be classified. We consider the question of distance measure, in general, and we will not address the problem of orientation. This can be handled in a preprocessing phase or incorporated into the distance measure if it is necessary. Let there be two images containing objects *A* and *B* and overlapped to each other, see Figure 3.3.

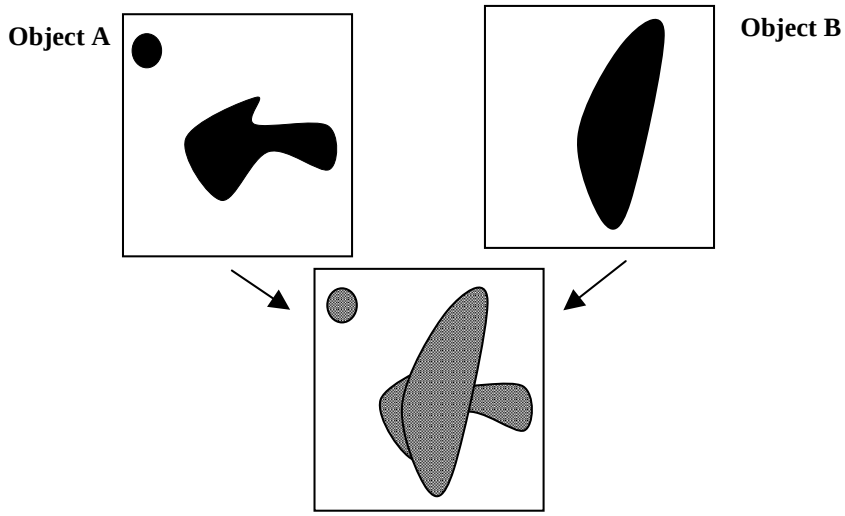


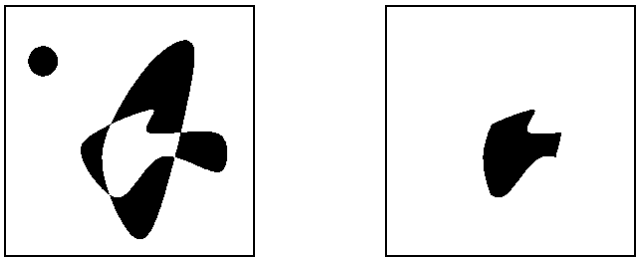
Figure 3.3 Two objects to be compared. Object A has two parts while object B forms a contiguous closed area. The objects are overlapped as the third image shows.

The most obvious criterion of the degree of coincidence of point sets is a measure of symmetrical difference (number of different points). This is a natural choice and it is the well-known Hamming distance (*HD*) which is the result of a pixel-wise XOR operation on binary images. Formally, the Hamming distance can be defined as

$$HD(A,B) = N\left[(A \cup B) \setminus (A \cap B)\right] \text{ or equivalently}$$
$$HD(A,B) = N\left[(A \setminus B) \cup (B \setminus A)\right] \qquad (10)$$

where *N[.]* means the number of pixels of the set. In case of previous objects *A*, *B* this is shown in Figure 3.4. If we consider binary images representing with 0 and 1 white and black pixels, respectively, then the Hamming distance equivalent to the Manhattan (city block) distance (Minkowsky with r = 1).

The Hamming distance is context free in the sense that each pixel is as important as any other pixel, independently of the position of the pixel. It is obvious that Hamming distance is sensitive to object shift and noise. Another problem is that Hamming distance cannot take into account the shape information because it measures only the area differences. The common part for a similarity measure would be important but it could also be incorporated into the distance measure via normalization.



HD = number of black pixels
a) Hamming distance b) Common part (intersection)

Figure 3.4 a) Hamming distance between A and B objects (*HD*) b) The intersection of A and B is not included in the measure.

The another often-used distance is the Hausdorff distance. Given two finite point sets *A* and *B*, the Hausdorff distance is defined as

$$HsD(A,B) = \max(h(A,B), h(B,A))$$
$$\text{where } h(A,B) = \max_{a\in A} \min_{b\in B} \|a-b\|, \text{ and } \|\cdot\| \text{ is some norm on the points of } A \text{ and } B. \tag{11}$$

This measure defines a distance function which is a metric over the set of all closed, bounded sets [34]. For an example see Figure 3.5 in which the definition is interpreted in such a way that the points are checked what vicinity of a point set they belong to.

The function *h(A,B)* is called the *directed* Hausdorff distance from *A* to *B*. It 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 ||.||). The *h(A,B)*, in effect, ranks each point of *A* based on its distance to the nearest point of *B* and then uses the largest ranked such point as the distance (the most mismatched point of *A*). Note, that in general *h(A,B)* and *h(B,A)* can attain very different values (the directed distances are not symmetric). The Hausdorff distance *HsD(A,B)* is the maximum of *h(A,B)*, and *h(B,A)*. Thus, it measures the degree of mismatch between two sets by measuring the distance of the point of *A* that is farthest from any point of *B* and vice versa. Intuitively, if *HsD(A,B) = d*, then each point of *A* must be within distance *d* of some point of *B* and vice versa. This metric is computable in polynomial time and the problem of computing the Hausdorff distance between geometric entities has been considered in the area of computational geometry (see e.g. [80], [81]). Unfortunately, this metric is not very well suited for some applications. The reason is that the Hausdorff distance does not take into account the overall structure of the point sets. Our aim will be to overcome this disadvantage. This issue will be discussed in the section 3.4.

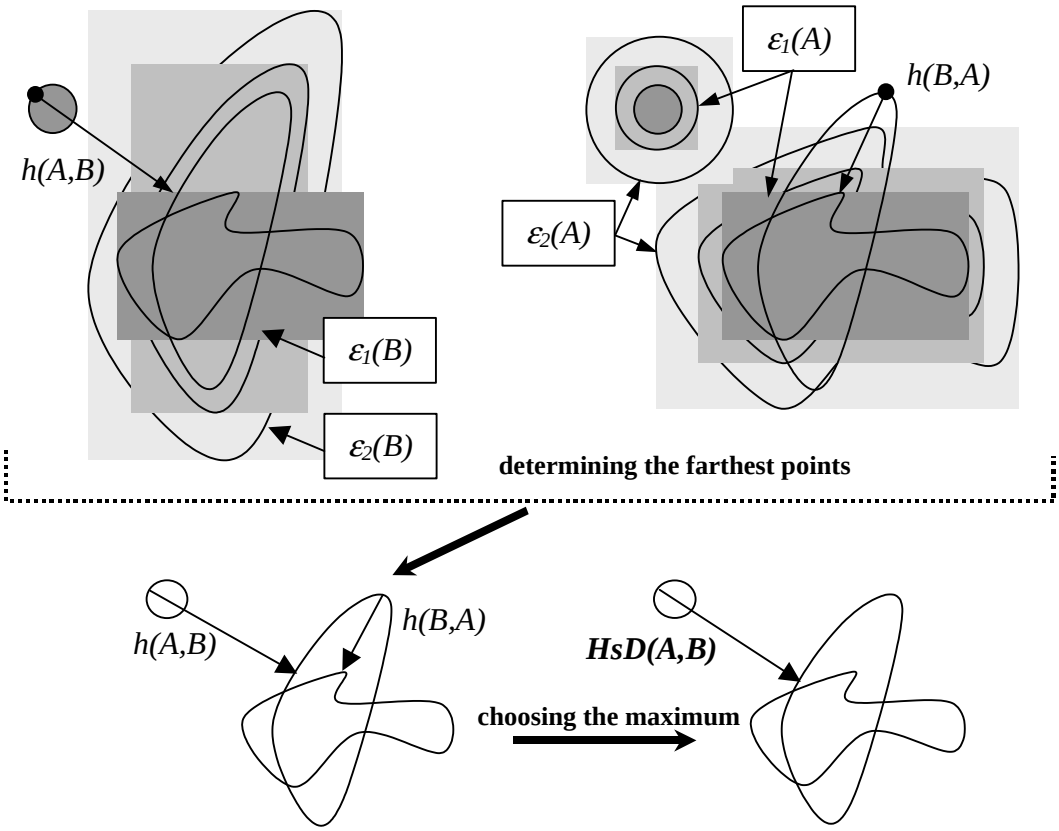


Figure 3.5 Hausdorff distance between A and B objects. Consider the ε-vicinity of set B, i.e. the points whose distances from B are not grater than ε (ε_1(B) is smaller than ε_2(B)). The minimal ε-vicinity of B which covers set A is the directed (asymmetric) Hausdorff distance of A to B (**h(A,B)**). The **h(A,B)** and **h(B,A)** are the directed (asymmetric) Hausdorff distances, while the **HsD(A,B)** is the Hausdorff distance as the maximum of the directed Hausdorff distances.

3.3 Nonlinear Hausdorff Metric (Autowave Distance) Computation on the CNN Architecture

In this section the implementation of the nonlinear Hausdorff distance on CNN architecture will be presented. As we will see both Hausdorff and nonlinear Hausdorff metrics can be implemented on CNN but the nonlinear variant is less noise sensitive and more suitable both from application and implementation point of view.

The Hausdorff distance measures the mismatch between two sets and shape similarity is not included straightforwardly. Unfortunately, Hausdorff distance is extremely unstable in the presence of noise. The appearance of a noisy spot, however small it can be, will drastically change the distance. For the sake of tolerance to noise, the definition of Hausdorff distance is modified in [86]. They use term *autowave distance* for this type of distance measure but thorough analysis showed that this is simply a nonlinear variant of the Hausdorff distance. Also the method of the measurement of the autowave distance introduced in [86] requires more investigation before a practical application.

The concept of the nonlinear Hausdorff distance is as follows. The distances between points are allowed to be measured only along paths wholly in $A \cup B$. Additionally, only those points are measured which are in $A \cup B$ and form a closed contiguous region with $A \cap B$. In other words, this distance is the Hausdorff distance associated with another metric of point space, namely, the metric generated by the union of A and B ($A \cup B$). That is, to compare images, the metric is used that itself depends upon the images compared, i.e. it is a "nonlinear" version of the Hausdorff distance. It is defined as

$$nHsD(A,B) = HsD(A`,B`) = HsD(A \cap B, A \cup B_{|\text{except incontiguous parts}}) = h(B`,A`) \quad (12)$$

Figure 3.6 shows these distances for objects A and B. The computation of the nonlinear Hausdorff distance is simpler than the Hausdorff because the directed Hausdorff distance should be performed only once instead of in the case of Hausdorff distance.

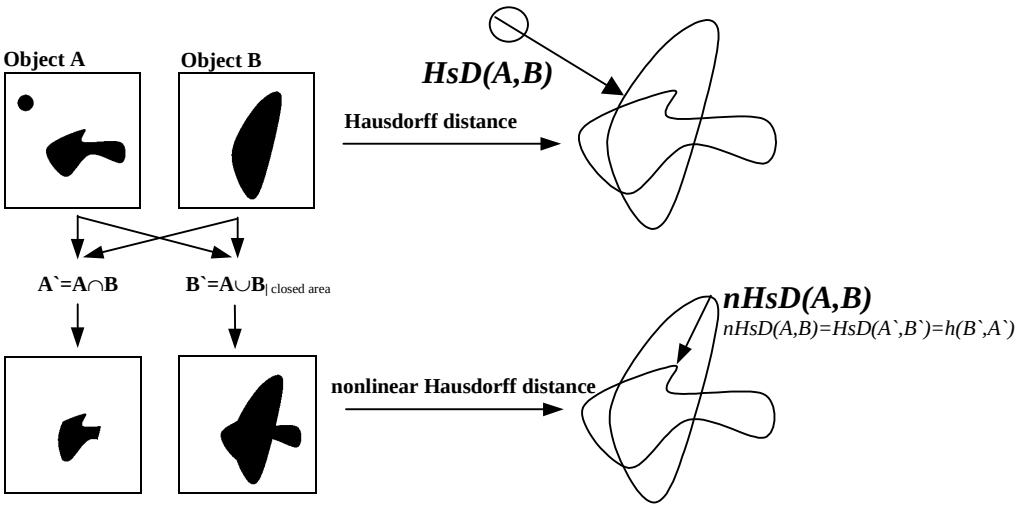


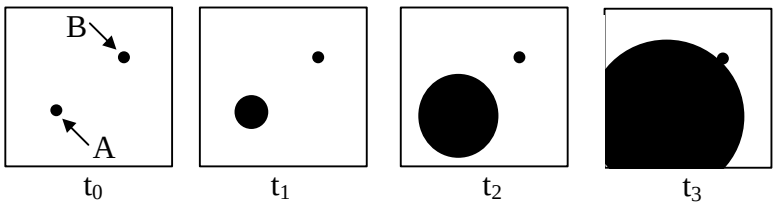
Figure 3.6 Hausdorff and nonlinear Hausdorff distances between A and B objects (*HsD*, *nHsD*). The nonlinear Hausdorff distance can be defined as the Hausdorff distance between the intersection and union of the closed area of the original objects.

The properties of nonlinear Hausdorff distance provide increased tolerance to noise effects than Hausdorff distance. For instance, the Hausdorff distance may be large depending on the position of the exceeding small circle in set A, whereas the nonlinear Hausdorff distance between these images does not depend on this. Nevertheless, both of the Hausdorff distance and its nonlinear version measure the mismatch between two sets and shape similarity is not taken into account.

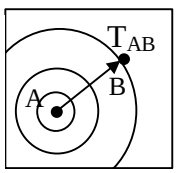
Even once a metric has been chosen, there are many ways of computing the distance transform of a binary image. Implementing the Hausdorff and nonlinear Hausdorff distance the synergetic approach is proposed in [86] where self-organization processes ([84-88]) are used for pattern recognition and other type of problems. The idea of that approach is to construct a system whose phase space would be related to the image space in a simple way, and to explore structure via propagating waves. The waves spread in active media at the expense of the energy stored in the medium. These waves are called autowaves where the activation of elements of the medium

propagates. The Hausdorff distance can easily be measured using a type of autowaves called trigger waves where the transition from inactive state to activated state of a cell propagate in the medium. Such measurement would require the generation of trigger wave, whose initial position coincides with one image. The wave propagates until all the points belonging to another image become triggered. As a norm, the time required to compute the operation is equivalent to the asymmetric (directed) Hausdorff distance. To measure the symmetric distance, this operation should be performed twice. Due to the property of nonlinear Hausdorff distance, this distance (called autowave distance in [86]) requires only one operation of wave propagation. Let the wave start to propagate from intersection of A and B ($A \cap B$) and spread only through the points belonging to the union A and B ($A \cup B$), instead of spreading everywhere. The time required for the wave to occupy $A \cup B$ can be used to define the measure of the difference between A and B.

But before this method can be applied there are some questions to be answered. What types of waves are good for such measurement? What types of controls of waves would be useful? What areas should be filled out? Time measure of propagation of waves can be treated as a norm? How can be this measure carried out? What type of system is suitable for an implementation? Next sections will discuss the implementation issue while now we would like to investigate the problem of use of waves for time measurement as a norm. We consider a system where the transition from inactive state to activated state of a cell can propagate in the medium. Suppose the task is already solved, in particular, the time measurement of wave propagation. If the system spatially continuous then using isotropic wave propagation the Euclidean norm, for instance, can be exactly calculated. Simple example demonstrates it in Figure 3.7. In a spatially discrete system (e.g. in a rectangular grid) the wave propagation can be isotropic in large scale, but in small scale this might be violated.



Snapshots of isotropic wave propagation from A to B.



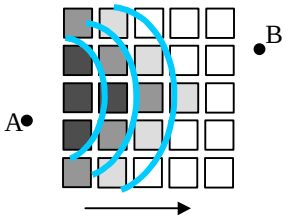
$L_2(A,B) \sim T_{AB}$
$L_2(A,B) = \gamma T_{AB}$

Time measurement is equivalent to Euclidean norm of spatial distance between A and B.

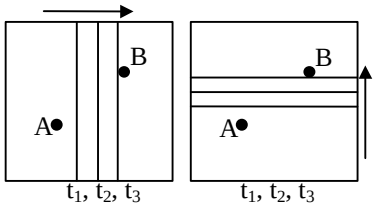
Figure 3.7 Continuous spatial representation of time measurement using wave propagation.

Figure 3.8 shows possibility how to compute Euclidean norm using two-step wave propagation. Horizontal and vertical wave front can propagate in isotropic way also in a spatially discrete system. Computing two time values for the horizontal and vertical directions then the Euclidean and Manhattan norm can be exactly calculated. However, we do not have to restrict ourselves to do this. To use wave propagation and its time measurement as a norm two conditions should be fulfilled, namely,

- *local condition*: transitions of cells should be strictly monotonic
- *global condition*: smooth wave front expending, i.e. collective expansion not a dendritic growth



Consecutive snapshots of wave front propagation in a spatially discrete system encoded by gray scale colors. In small scale the wave propagation is not isotropic. Therefore the Euclidean norm can only be approximated.



Using two-step wave propagation in horizontal and vertical directions the Euclidean (L_2) and Manhattan (L_1) norms can be exactly calculated.
$L_2(A,B) = \gamma_a L_2(T_x(A,B), T_y(A,B))$
$L_1(A,B) = \gamma_b L_1(T_x(A,B), T_y(A,B))$

Figure 3.8 Discrete spatial representation of time measurement using wave propagation.

Another problem where the wave front should propagate. Shall the wave propagate through the whole medium or only through the union of objects? Figure 3.9 shows these cases.

If there is a requirement to compare more objects on an image then it seems better to restrict the propagation of wave fronts only through the union of objects. In this case several object pairs can be compared at the same time and collisions of wave fronts can be avoided. Otherwise the measurement of time would not be valid because a wave front of an other object pair falsifies it. It should be noted that constraining wave propagation only through the union of objects might cause undesired effects in spatially discrete systems. For instance, if the objects details are comparable with the resolution of the system then speed of wave propagation might depend on the morphology of object. Therefore the time measurement will not be proportional, for instance, with the Euclidean norm. Nevertheless if this dependence is monotonic between propagation speed and object structure then the requirement of norm is still hold.

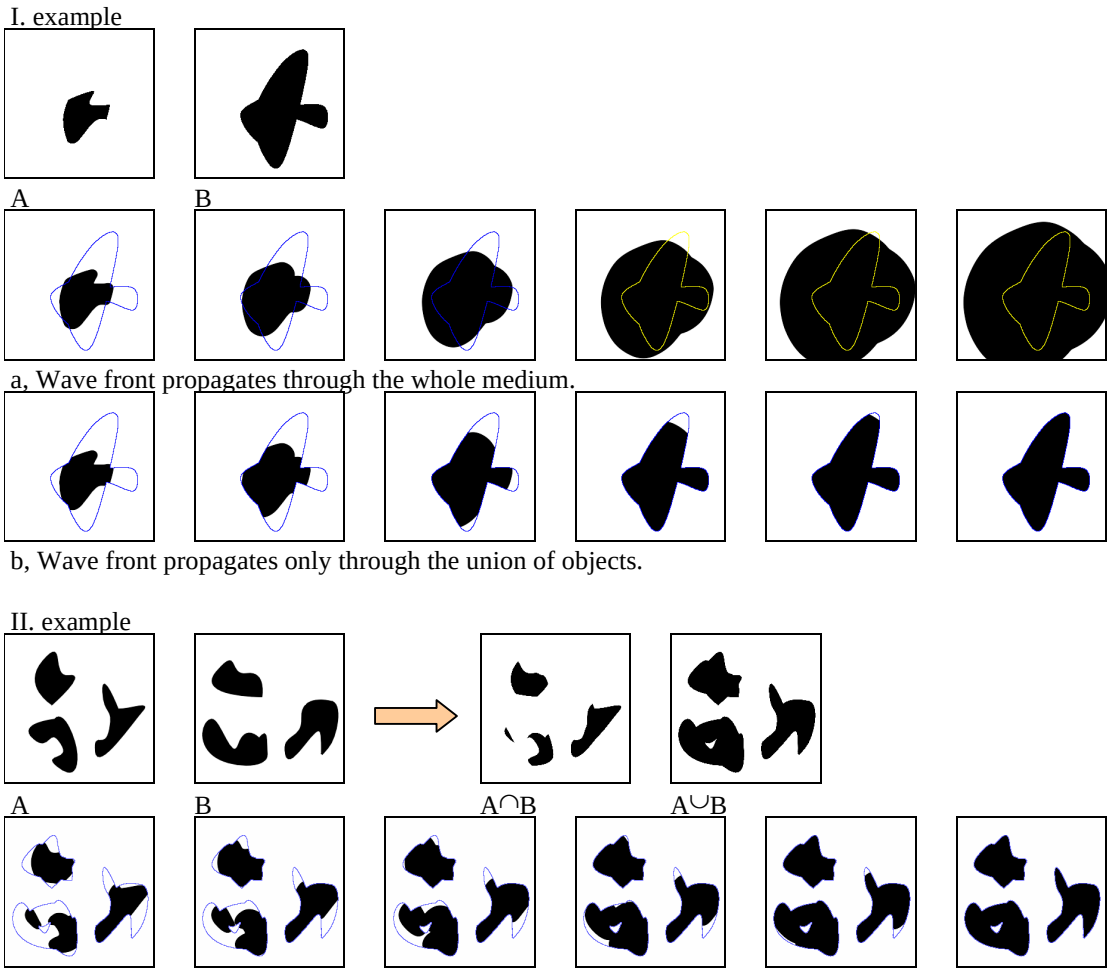


Figure 3.9 Wave propagation trough the medium depending on spatial constraints. If wave propagation takes place on more objects and propagation is restricted only through the unions of objects then collision of wave fronts can be avoided.

3.3.1 Implementing Nonlinear Waves on CNN

In this section the wave phenomena will be analyzed. The focus is put on the examination of wave generation especially the autowave approach will be discussed as a flexible method for pattern analysis. A special type of autowave called trigger wave will be used for time measurement.

3.3.1.1 Autowaves

The autowave approach can be used to solve different computational tasks formulated as image processing problems and it has several advantages especially for pattern recognition [84-88]. Other studies also used these phenomena as level set method by studies of Sethian [90, 91], front propagation by Osher [92], Malladi [93, 94], Rekeczky [21], flame propagation by Chorin [95], and deformable surfaces named as snakes by Kass [96].

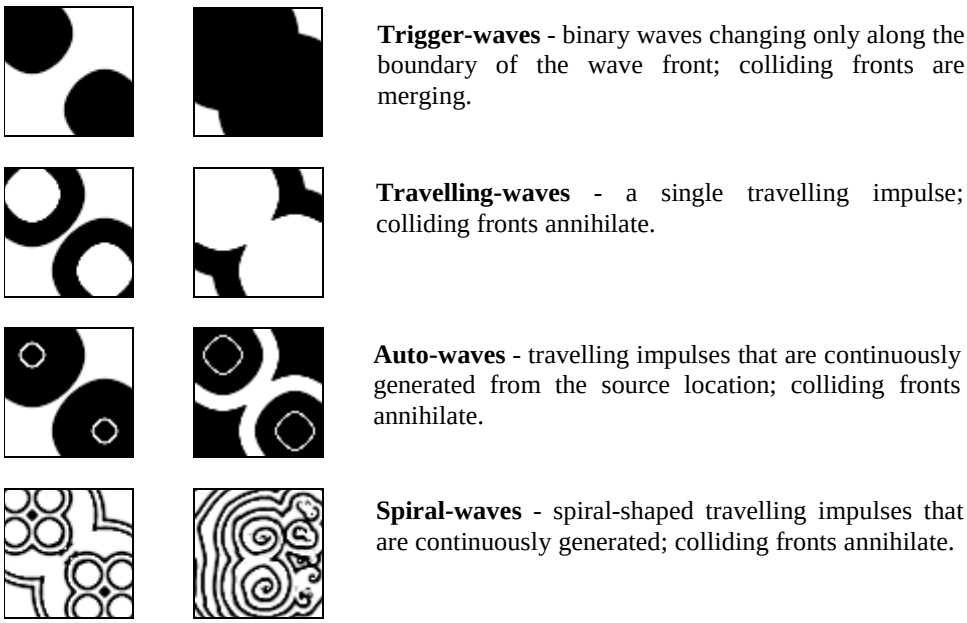
It is by now well known that numerous open systems in physics (fluids, plasmas, lasers, semiconductors), chemistry and biology may spontaneously develop spatial, temporal or spatio-

temporal structures by self-organization [84, 85]. Analogies between the corresponding patterns can be observed in spite of the fact that the underlying systems are of quite a different nature. For instance, in chemical reactions the typical reaction-diffusion equations are of the form of autowaves, which the focus will be put on. Autowaves represent a particular class of nonlinear waves which spread in active media at the expense of the energy stored in the medium [87, 88]. The properties of autowaves basically differ from those of waves in conservative systems, including nonlinear waves. The autowave, being a wave, can diffract and according to the Huygens' principle, bypass obstacles. However, it has unusual properties. Two waves spreading in opposite directions do not pass through each other, as is usual for classical waves, but they might mutually annihilate similar to particles. The fundamental properties of nonlinear waves in conservative systems and of autowaves are summarized in Table 3.1. It is worth to mention that concept of autowave is interpreted in many ways in the literature. Here, the concept will be used that a system governed by a reaction-diffusion process might produce autowaves.

Properties	Waves	Autowaves
Conservation of energy	yes	no
Conservation of amplitude and waveform	no	yes
Reversibility	yes	no
Reflection	yes	no
Annihilation	no	yes/no
Interference	yes	no
Diffraction	yes	yes

Table 3.1. Properties of waves and autowaves

In the class of autowaves belong several different types of waves. Some examples are shown in Figure 3.10.



Trigger-waves - binary waves changing only along the boundary of the wave front; colliding fronts are merging.

Travelling-waves - a single travelling impulse; colliding fronts annihilate.

Auto-waves - travelling impulses that are continuously generated from the source location; colliding fronts annihilate.

Spiral-waves - spiral-shaped travelling impulses that are continuously generated; colliding fronts annihilate.

Figure 3.10 Snapshots of different types of autowaves. They differ in some properties.

Autowaves can be described by a PDE of the form

$$\frac{\partial u}{\partial t} = D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + f(u) \tag{13}$$

Here, $\partial u / \partial t$ for an image, is the rate of change of intensity values *u(x, y, t)*. It is induced by the diffusion term $D(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2})$ and by the control function *f(u)*. Equation (13) describes an autowave if *f(u)* satisfies some requirements. It should have the form of a time varying or a nonlinear interaction.

Autonomous and non-autonomous CNNs provide an ideal framework for exploring wave phenomena and pattern formation [16-18]. It is worth to mention that nonlinear PDEs can always be regarded as a limiting form of CNNs (a proper spatial discretization always leads to a CNN). In simplest case it is a coupled first order nonlinear differential equation - ODE). Applications proposed for autowaves [88] can be realized by a CNN structure [19], [20], [21]. Autowaves were observed in a CNN array that have Chua's circuits as cells [22], [23]. There the nonlinear resistor of the chaotic oscillators provided the active local dynamics. Such a system can be built using a simpler CNN architecture with the original cell-type [24], [25]. There a single-layer architecture was shown where the active local dynamics were generated with a delay-type template which resulting in a bistable system. Here, the focus will be put on possible simplest trigger wave generation and its use for distance measurement. Trigger wave generation was thoroughly discussed by Rekeczky in [26].

3.3.1.2 Trigger Waves

For exploration of objects via propagating waves it seems to be the best solution if the colliding wave fronts do not annihilate. Otherwise such a situation might occurre that waves propagating along an object annihilate each other and measurement will produce false result. Since we do not need the annihilation property of autowaves for the present, we will focus on the simplest type of autowaves called trigger waves. A trigger wave, illustrated in Figure 3.10(a) is a binary wave that expands or shrinks along the boundary of two regions being in opposing state (in CNN terminology black represents +1 and white stands for –1 cell value). A trigger wave spreads at the expense of the energy stored in the medium (here the power supply of the CNN) and conserves the amplitude and wave form during the expansion. We need only these two properties. Trigger waves differs from autowaves in other two characteristics, namely, they do not annihilate and have the reversibility property. By reversibility we mean the inversion of the motion direction of the wave front. It should be mentioned that in this case the conversation of wave front might be violated if the sharp extensions (high frequency components) of an object are smoothed out from the shape during the propagation. It means that reverting simply the propagation of the trigger wave will not reconstruct the object. In general, the trigger wave propagation is a nonlinear mapping that is not invertible. This is not necessary a disadvantage, since in some application smoothing the boundary might be useful process. The main properties of trigger waves are shown in Figure 3.11.

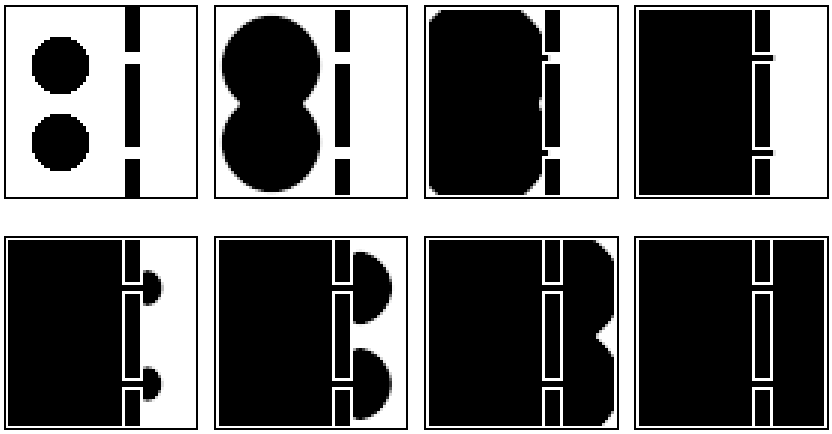
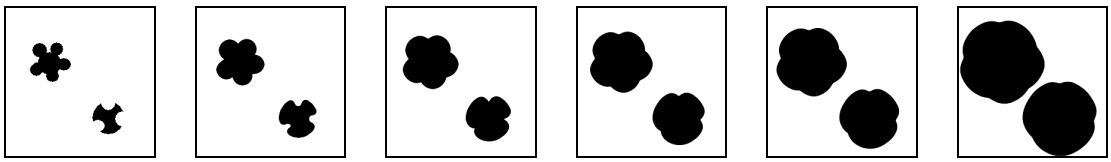
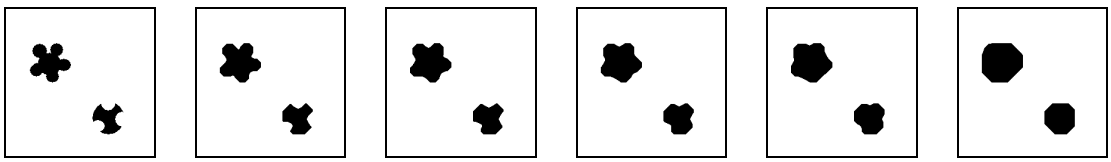


Figure 3.11 Demonstration of trigger-wave properties near an obstacle and the collision of two wave-fronts. Trigger-waves do not reflect from the obstacle and from the boundaries conserving their wave form and amplitude. During the collision of two wave fronts no interference can be observed, these waves tend to merge (Remark: cells around the obstacle and the boundary cells were fixed).

To generate a trigger wave the CNN must work in a local diffusion mode (Thiran and Setti [16], [17]). Local diffusion mode means that only locally neighboring cells can affect each other. This is related to the diffusion part in Equation 13. As a consequence we should differentiate terms *local* and *global propagation*, what comes from the geometrical description of wave propagation. A trigger wave is said to be *globally propagating* if locally convex black region can trigger neighboring cell in white region and vice versa. Thus the black region is globally expanding. A trigger wave is *locally propagating* if changes occur only if locally concave black regions trigger a cell then propagation must stop after some time when all these locally concave regions have been triggered. This is demonstrated in Figure 3.12.



a, global propagation of trigger wave front



b, local propagation of trigger wave front

Figure 3.12 Global and local propagation of trigger wave fronts. Trigger wave is globally propagating if convex black regions can trigger places in white region (and vice versa). It means that without any additional constraints black region will expand through the whole medium. A trigger wave front is said to be locally propagating if concave regions can only trigger white regions. Thus the propagation must stop after some time when all these locally concave regions have been triggered.

In our case, only globally propagating trigger waves are acceptable because we need propagation on objects with no restriction otherwise time measurement will be wrong. Two types of implementation will be discussed. One of them generates waves propagating in the system in a single transient therefore this is referred as continuous time approach. An other one constructs

waves in an iterative process where binary morphology operations are used. This method is a discrete time approach from algorithmic point of view although the basic operations are continuous too. The second approach will be used as element tool for time measurement on present CNN chips.

3.3.1.3 Single Transient Trigger Wave Generation (Continuous Time Approach)

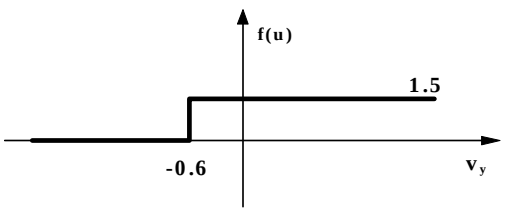
Here, the possible simplest solution was chosen for trigger wave generation, namely, waves propagate on a single-layer architecture with the original cell-type and the active local dynamics are generated with a simple nonlinear function. This takes into account the problem of the VLSI implementation. By proper discretization of Equation 13 we obtain:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{dt}}\mathrm{u}(\mathrm{x},\mathrm{y},\mathrm{t}) &= \mathrm{D}\left(\frac{\partial^2\mathrm{u}(\mathrm{x},\mathrm{y},\mathrm{t})}{\partial\mathrm{x}^2}+\frac{\partial^2\mathrm{u}(\mathrm{x},\mathrm{y},\mathrm{t})}{\partial\mathrm{y}^2}\right)+\mathrm{f}(\mathrm{u}) \\ &\approx \mathrm{D}\frac{1}{4}\left(\mathrm{u}_{\mathrm{ij}-1}(\mathrm{t})+\mathrm{u}_{\mathrm{ij}+1}(\mathrm{t})+\mathrm{u}_{\mathrm{i}-1\mathrm{j}}(\mathrm{t})+\mathrm{u}_{\mathrm{i}+1\mathrm{j}}(\mathrm{t})\right)-\mathrm{Du}_{\mathrm{ij}}(\mathrm{t})+\mathrm{f}\left(\mathrm{u}_{\mathrm{ij}}\right)\end{aligned} \tag{14}$$

The autowave equation can be directly mapped onto the CNN array (D=1) resulting in the following simple template

$$\mathrm{Trig}_1 : A=\begin{bmatrix}0 & 0.25 & 0\\ 0.25 & 0 & 0.25\\ 0 & 0.25 & 0\end{bmatrix},\qquad \hat{A}=\begin{bmatrix}0 & 0 & 0\\ 0 & f(u) & 0\\ 0 & 0 & 0\end{bmatrix},\quad B=[0],\quad z=0$$

In the middle of template *A(.)* the effect of *-1/R* in the CNN equation is considered (*R=1*). What is the proper control function *f(u)* for trigger wave generation? As the term *trigger* indicates that during the diffusion process the cells should be switched on when they reach a given value. So the simplest control function providing a threshold dependent switch can be, for instance,



The initial state should contain the trigger points of the trigger waves. The advantage of this template is that the speed of the wave propagation can be adjusted by the maximum and the threshold position of *f(u)*. Here, it is not our aim to discuss thoroughly different possible types of the nonlinear control function. Or focus is put on trigger wave generation only. Although the *f(u)* is the simplest nonlinearity useable for trigger waves it is still not available in existing CNN chips. Taking into account that present CNN chips does not support nonlinear interaction, we give a linear template for trigger wave generation. The output nonlinearity of the CNN cell (sigmoid type cell output) makes possible the implementation of the control function without nonlinear cell interaction.

$$\mathrm{Trig}_2 : A=\begin{bmatrix}0.41 & 0.59 & 0.41\\ 0.59 & 2 & 0.59\\ 0.41 & 0.59 & 0.41\end{bmatrix},\ B=[0],\ \ z=4.5$$

The template entries and bias term can be obtained if the CNN equation is solved using the condition that even one black (+1) pixel surrounding with white pixels should propagate. The concrete values are modified with proper diagonal weighting. The initial state contains the trigger initials of the waves. Based on theoretical consideration, the elements of *A(.)* template can be chosen from a wide interval providing the necessary robustness for an analog implementation. Its advantage is the easy implementation and already can be tested on CNN chips. The speed can also be adjusted through the bias term although it effects some changes in its properties (e.g. the number of black neighbors that is needed to exceed the activation). It means, for instance, that changing the bias term below a value (here $-1< z <1.5$) then property *global propagating* will be lost and the template supports only *local propagating*.

General template design techniques were discussed in Zarandy's work [31]. The problem of trigger wave generation has been treated in a systematic way by Rekeczky in [26]. In his work, he analyzed symmetric and isotropic templates in standard continuous-time CNN. Time domain analysis, local statistical analysis, local geometrical analysis, and spectral domain analysis were thoroughly discussed. A number of intuitively developed templates from the template library [32] has been grouped in a well-defined frame and it has been shown that qualitative properties of trigger wave generation templates can be simply controlled through the bias term of the CNN equation.

3.3.1.4 Iterative Trigger Wave Generation (Discrete Time Approach)

Although an iterative trigger wave implementation is more time consuming solution than a continuous one but this is the only feasible way for implementing time measurement on the present CNN chips. The term of discrete time approach indicates that wave is generated in an iterative process propagating waves step by step. One step is executed by applying a basic binary morphology operation called dilation [28]. This operation is defined on two binary images, one being the operand, the other the structuring element. In the CNN implementation, the former is the input, while the function (template) itself depends on the latter. When calculating the operator, the image should be put to the input of the CNN, and the initial condition is zero everywhere. Intuitively, if the structuring element (template) is specified as below then each step of the iterative process expands objects width one pixel.

$$\mathrm{Trig}_3 : A = [2],\quad B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},\quad z = 8.5$$

This can be considered as one snapshot of wave propagation. For the next step of the iteration the output of the CNN should be fed back to the input.

3.3.1.5 Hardware Requirements of Wave Generation

This section discusses the implementation of trigger wave generation in a CNN Universal Chip environment. Several CNNUM architectures have been fabricated by many research groups [11-14] but up to now the truly operational CNNUM with large number of useful functionality is the 64x64 chip called ACE4K was designed in Seville [12]. The focus will be put on this chip. This chip contains 4096 analog programmable processor elements with analog, binary and optical inputs and analog and binary output. Almost one million transistors work in analog

domain. The chip contains on-chip 4 grayscale image memories (LAM), 4 binary image memories (LLM), 32 template memories (TEM), and 64 stored switch configurations (SCR) enables to execute long and complex image processing tasks.

To implement trigger wave templates on this chip some modifications are necessary in the previous templates. Because linear interactions are only supported the *Trig*$_2$ and *Trig*$_3$ templates can be used. The chip realizes the *Full Range Model* in which there is no linear resistance connected parallel with the cell capacitance therefore the CNN equation does not contain the *–1/R* term. So the central element of the *A(.)* template should be less with one. The two templates for continuous and discrete time approaches derived from *Trig*$_2$ and *Trig*$_3$ templates are the following.

$$\mathrm{Trig}_4 : A = \begin{bmatrix} 0.41 & 0.59 & 0.41 \\ 0.59 & 1 & 0.59 \\ 0.41 & 0.59 & 0.41 \end{bmatrix},\ B = [0],\ z = 5.0 \qquad \mathrm{Trig}_5 : A = [1],\ B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},\ z = 5$$

Both the LAMs and LLMs can be used for trigger propagation. Figure 3.13 shows an example for trigger wave propagation on the ACE4K CNNUM chip.

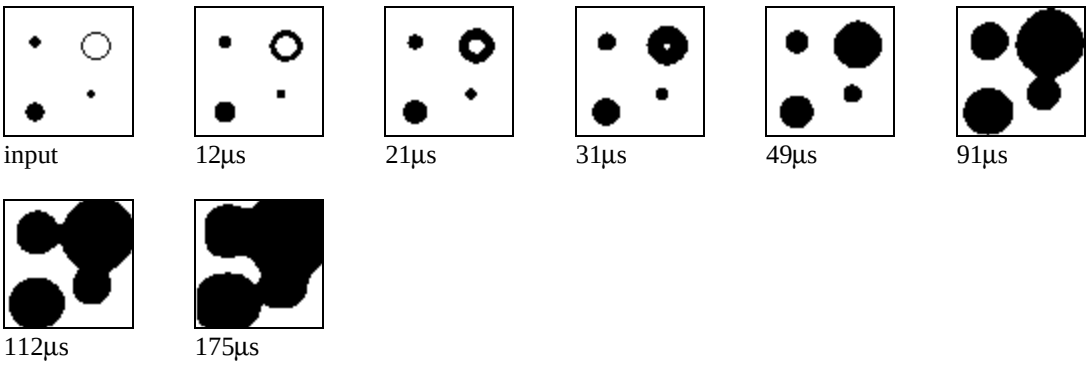


Figure 3.13 Trigger wave propagation on the ACE4K CNNUM chip.

The propagation depends on settings of wave generation. This might cause some undesired effects. For instance, the speed of wave propagation strongly depends on widths of objects where wave propagates as it is shown in Figure 3.14. This dependence always occurs due to the diffusion term in Equation 13. This effect can be strongly reduced if the transition of a cell from one state to the opposite is very fast, i.e. the dynamic of the template is as high as possible. This is constrained by the chip. The Trig4 and Trig5 are ideal from this viewpoint.

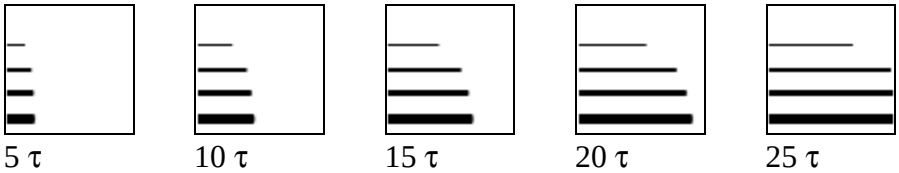


Figure 3.14 The speed of wave propagation depends on settings of wave generation. In example, image size is 64x64 and lines widths are 1, 2, 3, and 5 pixels.

Nevertheless this is still good for time measurement as we mentioned before that it is enough to assure the monotonic transition for a cell.

3.3.2 Implementing the Nonlinear Hausdorff metric (Autowave Distance) on CNN

Two types of implementation will be discussed. One of them is a straightforward method called dynamic solution the another one is an iterative solution. The VLSI complexity of the first method is sophisticated, and current CNN chips do not support, nevertheless this one takes all advantages of the CNN dynamics. The second one is more sophisticated from algorithmic point of view but requires only binary morphology operations implemented with linear templates on CNN and can be tested on present CNN chips.

It should be emphasized that the same technique can be used to implement Hausdorff metric on CNN architecture. As we have seen from the previous consideration, the nonlinear Hausdorff metric is more noise redundant therefore only its implementation will be detailed.

3.3.2.1 Dynamic Solution

Below we discuss the dynamic solution in details. Figure 3.15 shows a possible implementation of the nonlinear Hausdorff metric on two-layer CNN. The advantage of this implementation is that several object-model pairs can be compared at the same time. The time consumption of this implementation depends on the network size, i.e. the area where trigger waves might propagate. Nevertheless, if this algorithm will be used on real CNN-UM chips this will yield a very fast computation of the nonlinear Hausdorff metric.

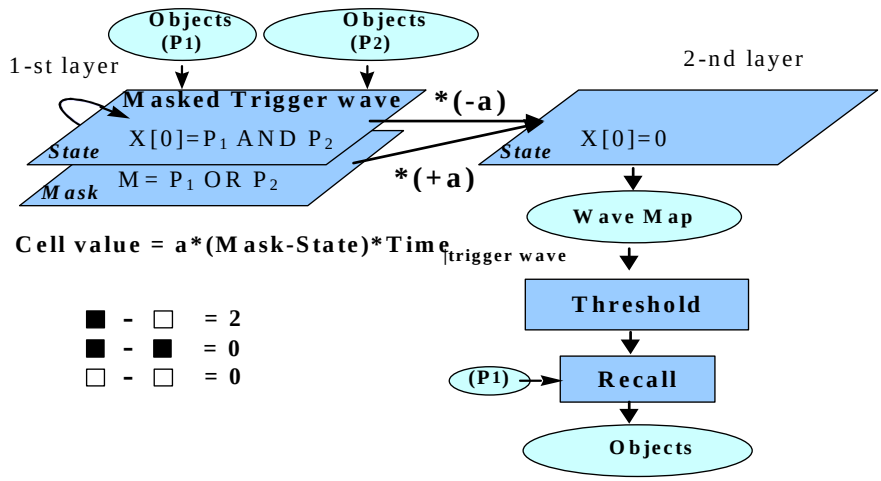


Figure 3.15 Dynamic implementation of nonlinear Hausdorff metric on CNN. From the intersections of sets to be compared trigger waves propagate on the first layer and time is measured via constant current filling on the second layer. The current term has only three possible combinations.

Two images will be compared. The first image contains the objects and the second one contains their models. The idea is that trigger wave will spread out from intersections of corresponding objects and models and the time is measured while wave occupies the union of objects and models. The union of sets is used as a mask to control the trigger wave enabling the propagation of wave only through the union. The initial points of the waves are the cells of the intersections of sets to be compared. At the time while waves are propagating on the first layer

the cells in the second layer are filled with a constant current to measure the time. The current has two components. One of them is determined by the mask the other one comes from the layer where trigger waves propagate. Due to the binary wave propagation the current is constant 2a at a given cell iff the mask is black but the wave has not reached yet this position otherwise it is zero. At the end of the process the cells at boundaries of the unions of objects and models will contain the highest voltage levels. Thresholding this image at appropriate level gives the nonlinear Hausdorff distance. At last those objects might be recalled where these differences are large indicating large difference.

3.3.2.2 Iterative Solution

The iterative method for nonlinear Hausdorff distance computation is presented in Figure 3.16. Its advantage is that it can be tested on present CNN chips although it is a time consuming algorithm. The number of iterative steps depends on object's sizes to be compared.

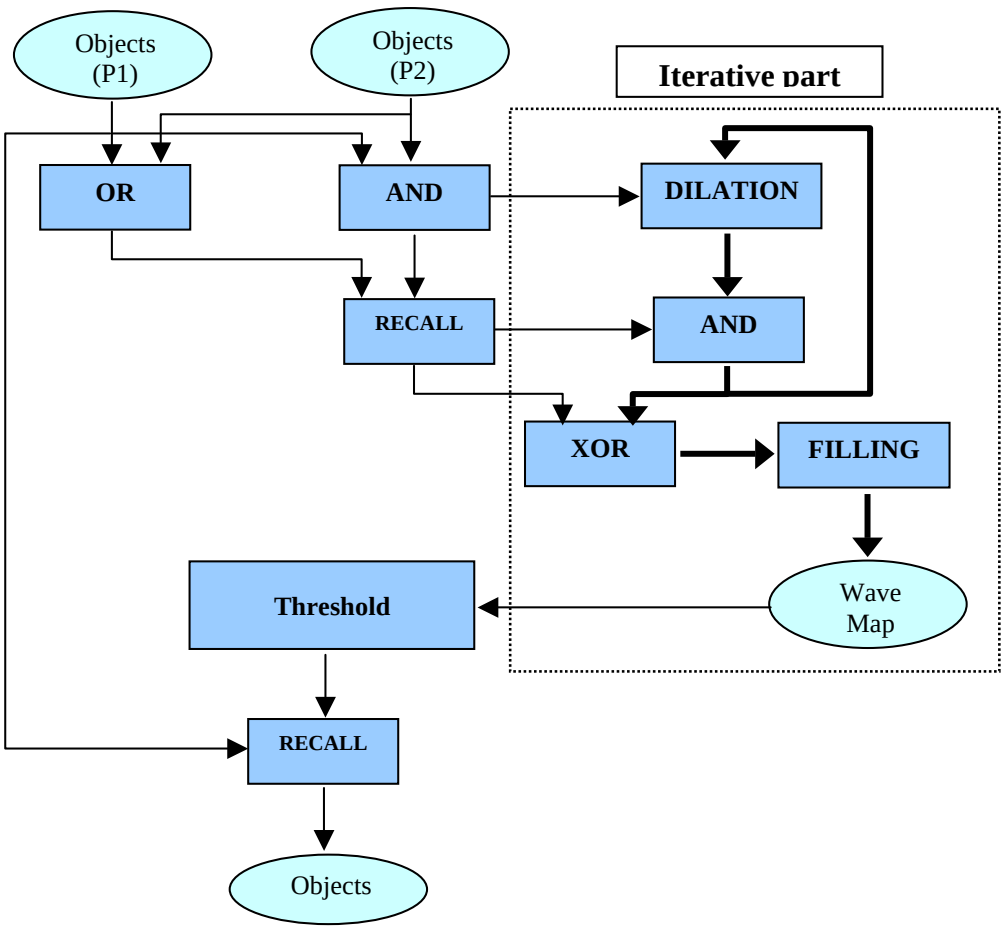


Figure 3.16 Iterative implementation of nonlinear Hausdorff metric on CNN. From the intersections of sets to be compared a binary morphology operation namely dilation expands the objects width one pixel in each step. The result of a step is used to fill a layer with a constant current for a given time producing quantized map. At the end of the process the complete wave map is obtained.

3.3.2.3 Hardware Requirements of Dynamic and Iterative Solutions

We have described two possible implementations of nonlinear Hausdorff metric on CNN that can be applied to object segmentation and classification. The main advantage of this implementation is that the time consuming Hausdorff distance can be computed very fast and opens the possibility to use this distance computation in real time application without very expensive huge computers.

The discussed dynamic solution requires the so-called fixed-state map technique, nonlinear cell interactions, and it is a two-layer approach. The VLSI implementation complexity of the solution mainly depends on the implementation of wave generation, since this is the only building block which require nonlinear template interaction. The dynamic solution will be tested on the next generation a CNNUM chips [27], while the iterative approach can already be tested on present CNNUM chips.

3.3.3 Examples using nonlinear Hausdorff metric

Two examples will be presented demonstrating some advantages of the (nonlinear) Hausdorff distance over the Hamming distance. Here, it is not our aim to discuss potential applications for (nonlinear) Hausdorff metric. Anyway, one possible application will be presented in Chapter 4 where a classification problem will be dealt with by using this metric. In both examples presented in Figure 3.17 and 3.19 it is demonstrated that a classification decision depends strongly on the used distance calculation. In Figure 3.17 we compare a circle shape object (**m**) as a model to three other objects. Two of them are also circles (**o1**, **o3**) while the third (**o2**) is “very” different for a human observer. Nevertheless, the Hamming distance ranks this object to be closer to the object **m** than others while the (nonlinear) Hausdorff metric chooses two circles closer to the model **m**, although the object **o3** is not only larger than **m** but also shifted simulating the position error.

In Figure 3.18 the time development of the nonlinear Hausdorff distance is shown in case of object **o1** and **o2**. As it can be seen, although the number of pixels to be filled is higher of object **o1** than **o2** the wave front covers it faster because the main amount of difference is near to the reference object (**m**) therefore the Hausdorff distance will be smaller. This is the reason behind the result why the Hausdorff distance ranks object differently.

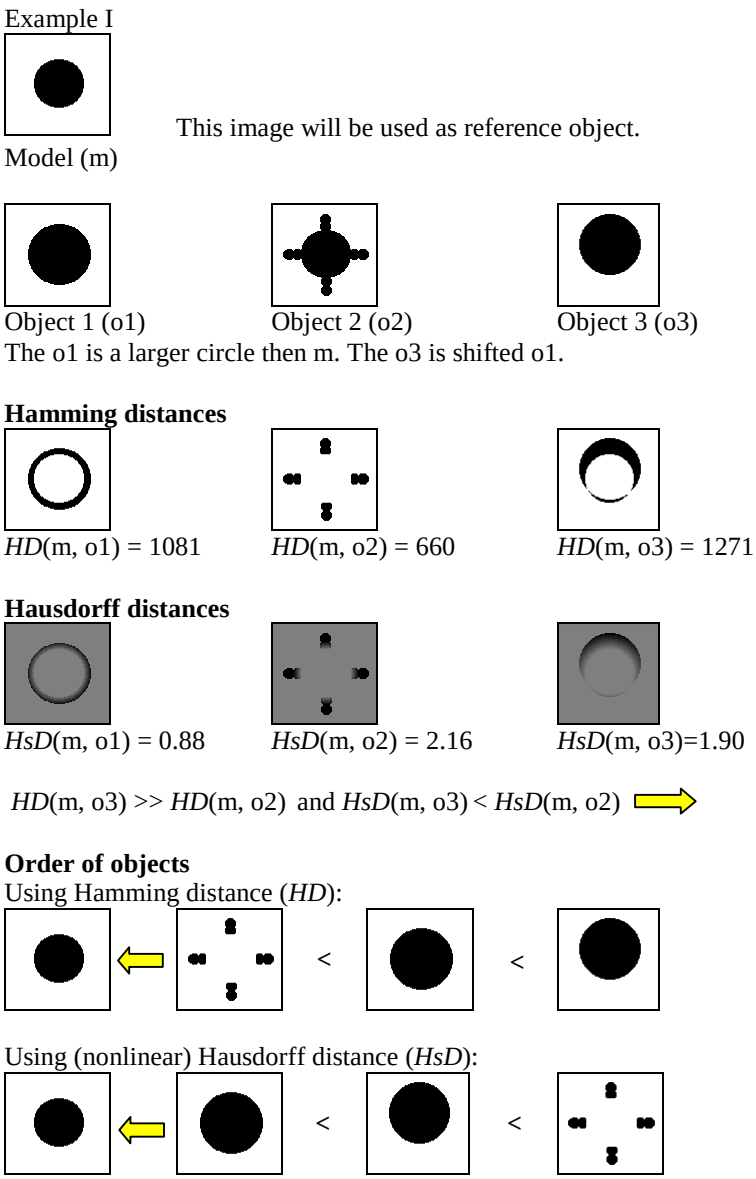


Figure 3.17 Using different distance metrics for object matching. Object **o1** and **o3** resemble more model **m** than object **o2**. Objects are positioned in case of **o1** and **o2** so the Hamming distances result in lowest values. The object **o3** is identical with object **o1** but it is shifted resulting much larger Hamming distance. The (nonlinear) Hausdorff metric chooses object **o1** like a human observer would and object **o3** is also closer to object **m** than **o2**.

Time development shown in Figure 3.18 will play very important role in metric assessment. Its variant will be used as a distribution function and will serve to compute different metrics (see e.g. Figure 3.25 and 3.44).

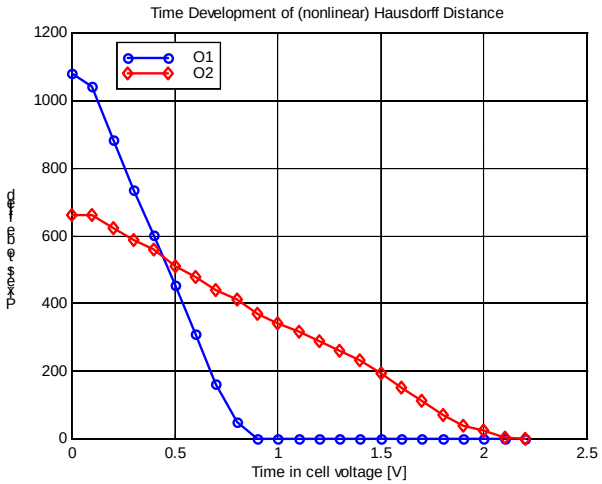


Figure 3.18 Time development of nonlinear Hausdorff distance in case of object **o1** and **o2**. The unfilled pixels are plotted versus time during trigger wave propagation.

In the second example (Figure 3.19), the effect of the choice of metric is demonstrated in case of letter classification. The type of letters is Times New Roman. Not all letters from the alphabet are presented here but the analysis showed that for errors like in figure (missing parts or needles extensions) the (nonlinear) Hausdorff metric based classification is more suitable than the Hamming distance.

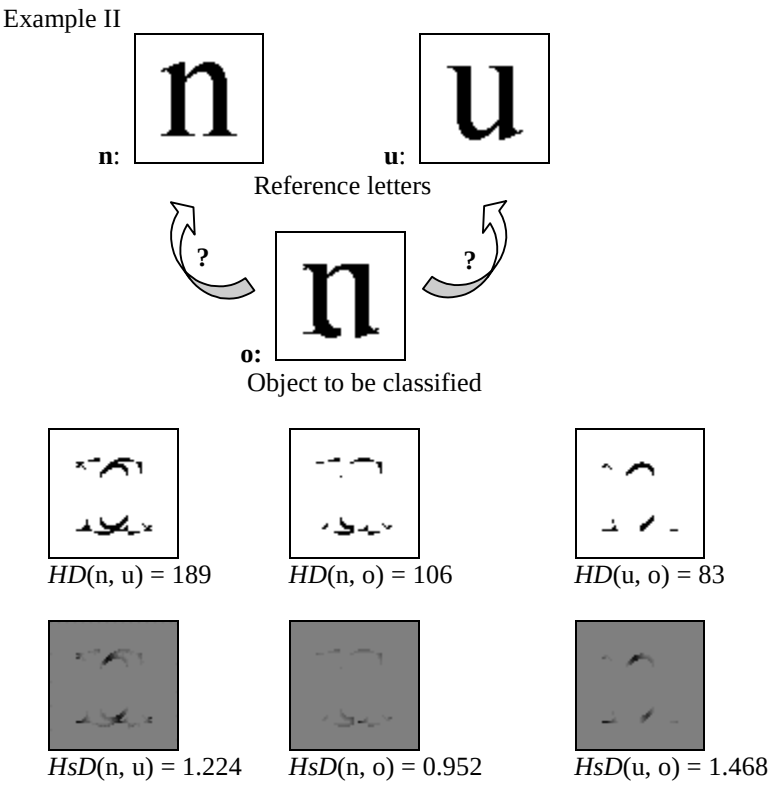


Figure 3.19 The (nonlinear) Hausdorff metric classifies object **o** correctly to object **n** as a letter n, while object **u** is closer to object **o** in Hamming distance sense.

It might seem that the (nonlinear) Hausdorff distance is more appropriate for object comparison than the Hamming distance but this cannot be stated so simple. There might be cases where Hamming distance is better than the Hausdorff distance. But there are cases where none of them can solve the given task. This subject will be discussed in the next chapter.

3.4 Introducing the Weighted Hamming (Integrated Hausdorff) Metric

Although the Hamming and Hausdorff distances are commonly used in image processing applications for object comparison and classification, they have several disadvantages. This section will investigate this issue and show their constraints. Based on the results of the previous section a novel type of distance computation will be introduced that can overcome these disadvantages. The proposed metric can be computed efficiently on parallel architecture therefore CNN type implementation of this metric provides a qualified tool for this type of comparison.

3.4.1 Constraints of Commonly Used Metrics

Here, a simple example will be shown to demonstrate the limitations of the presented metrics, see Figure 3.20. Two binary images will be compared. One of them contains a simple rectangular shape the another one contains a Gauss function and the area between the curve and the x-axis is filled. The discussed metrics are examined as a function of parameters of the Gauss function. The advantage of this example is that we can compute metrics analytically using the assumption that the image space continuous. The concrete images are, of course, limited and discretized in space but the measured distances give good approximation of the continuous case.

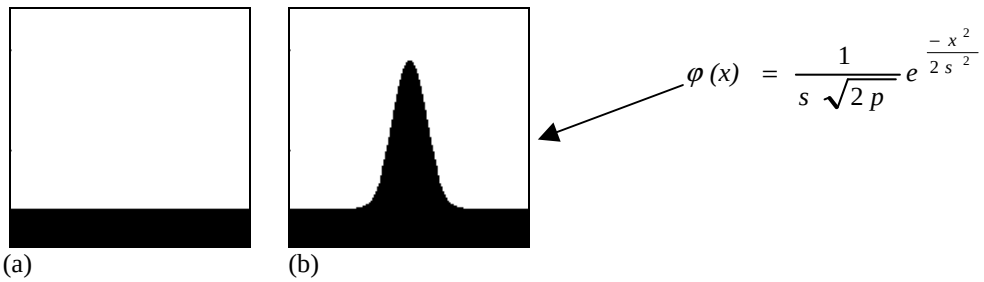


Figure 3.20 Example to demonstrate the limitations of different metrics. (a) contains the reference object while (b) is an image series where object properties are modified via function parameters. Image size is 256x256.

Hamming distance (the filled area) ~ $\Phi(x) = \int_{-\infty}^{+\infty} \varphi(x) = 1$

Hausdorff distance/Nonlinear Hausdorff distance ~ $\max(\varphi(x)) = \varphi(0) = \frac{1}{\sigma\sqrt{2\pi}}$

In this example the Hausdorff distances and Nonlinear Hausdorff distances are equal because objects form contiguous areas. Three cases will be considered, namely,

Function	Hamming distance	Hausdorff / Nonlinear Hausdorff distance
$f(x,c)$	$\int_{-\infty}^{+\infty} f(x)$	$\max(f(x))$
1. $f(x)=c\cdot\varphi(x)\rightarrow$	$\int_{-\infty}^{+\infty} f(x)=c\cdot\Phi(x),$	$\max(f(x))=c\cdot\max(\varphi(x))$
2. $f(x)=\varphi_\sigma(x)\rightarrow$	$\int_{-\infty}^{+\infty} f(x)=\Phi(x),$	$\max(f(x))=\frac{1}{\sigma}\cdot\max(\varphi_1(x))$
3. $f(x)=\varphi(c\cdot x)\rightarrow$	$\int_{-\infty}^{+\infty} f(x)=e^{c^2}\cdot\Phi(x),$	$\max(f(x))=\max(\varphi(x))$

Figure 3.21 illustrates the three different cases and distances measured to the reference object. In the first case both the Hamming distance and Hausdorff and nonlinear Hausdorff distances are linear function of the *c* parameter. In the second case the Hamming distance is constant, while the Hausdorff and nonlinear Hausdorff distances depend on the σ parameter. Since the Gauss function describes a probability function the integral (filled area) is constant therefore the hamming distance is constant. In the third case the Hamming distance depends on the c parameter, while the Hausdorff and nonlinear Hausdorff distances are constant since the maximum of the function does not change.

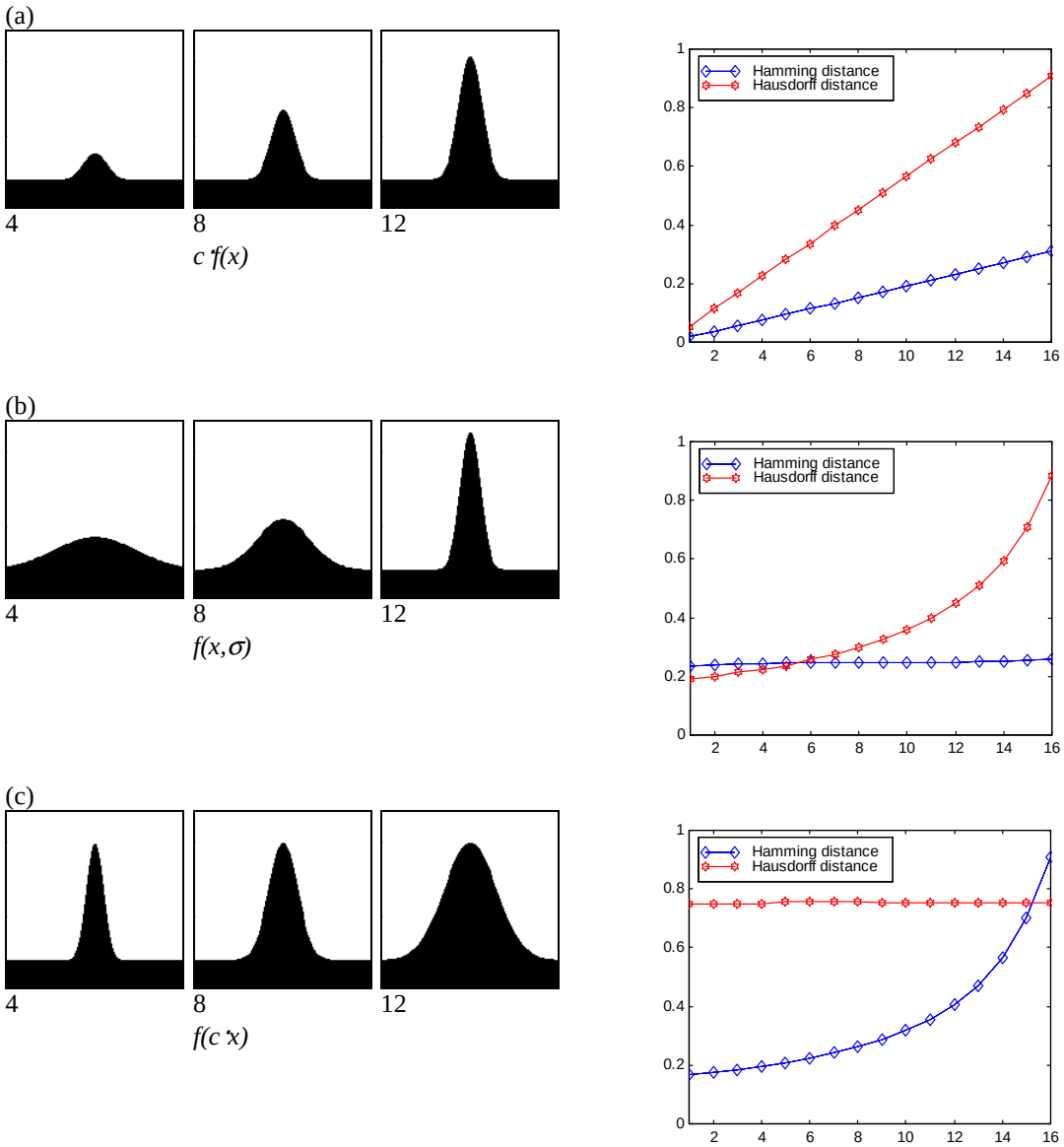


Figure 21 Example to demonstrate the limitations of different metrics. Three different cases of object series and results of their measurements are presented. The horizontal axis shows picture indices and the vertical axis shows the normalized distances.

The pictures are very different, nevertheless, the metrics cannot separate them in all the three cases. What are the reasons behind this failure? One of the disadvantages of the Hamming distance is that it cannot separate equal area differences. It measures only the “area or mass difference”, but does not contain information about the properties of this difference. Each different pixel gives same contribution to the distance but there is no grading among pixels. If a Hamming distance is given we can imagine many possible shape and their distortions that can be result of this metric computation (see Figure 3.22).

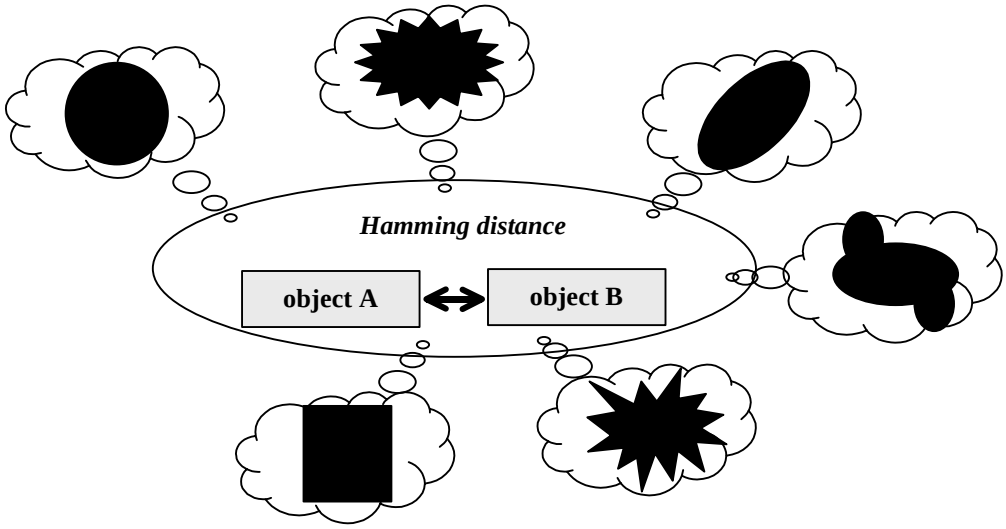


Figure 3.22 Computing the Hamming distance between two objects does not carry any information about the properties of this difference except the number of different pixels. Several shapes can produce the same distance although their characteristics might be very different.

The disadvantage of Hausdorff and nonlinear Hausdorff distance is that they cannot separate differences having peaks with equal length. The Hausdorff and nonlinear Hausdorff distances take into consideration only the farthest point between two sets and do not measure another points (see Figure 3.23).

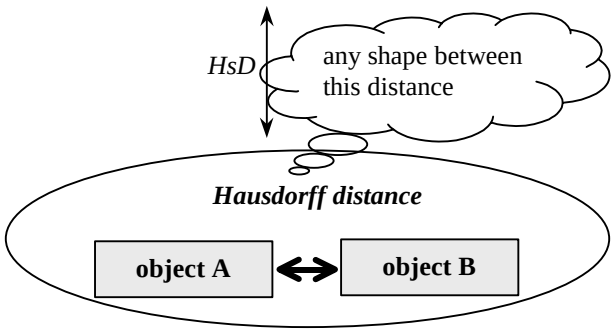


Figure 3.23 Computing the (nonlinear) Hausdorff distance between two objects does not carry any information about the properties of this difference except the distance of the farthest point. Several shapes can produce the same distance although their characteristics might be very different.

The main properties of the Hamming and (nonlinear) Hausdorff distances are summarized in Table 3.2.

Distance metric	What is taken into consideration?	What is the rating?
Hamming	each point	there is no grading
Hausdorff (nonlinear Hausdorff)	single point (farthest)	the most important

Table 3.2 Properties of the Hamming and Hausdorff metrics. Each metric measures a specific property.

These two metrics can be characterized as two extremities. The Hamming distance measures each difference without grating while the Hausdorff distance measures only one point although that point is the most important.

Figure 3.24 shows simple patterns where these metrics fail to distinguish objects comparing to a basic object. The first two examples (a), and (b) are artificial while the third (c) contains simplified images of propellers. Using a reference object (circle or pentagon in the middle of patterns) neither the Hamming nor the Hausdorff distance can measure the differences because objects are equidifferent in area or in farthest point.

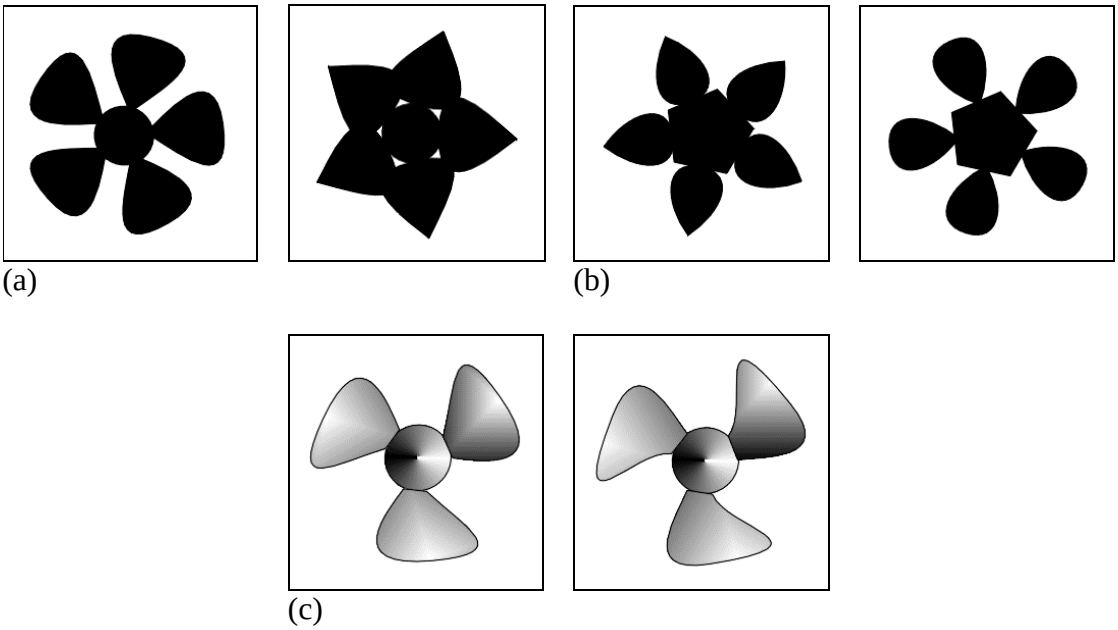


Figure 3.24 Simple geometric patterns demonstrate cases where Hamming and Hausdorff metrics fail. In the three cases, (a), (b), and (c) objects are compared to their central basic object (circle or pentagon). The computed distances are same because there is no difference in area or in farthest point.

3.4.2 The Weighted Hamming (Integrated Hausdorff) Distance

As we have seen in the previous section, there is a need to extend the difference measurement both in case of Hamming distance, Hausdorff, and nonlinear Hausdorff distances. Can we construct such a metric, which fulfills these requirements? The idea of the new metric comes from two thoughts. Why do not we combine the Hamming and Hausdorff distances together? We could measure each different point between two objects and rating them as the Hausdorff measure does. This is in some sense the choice of the golden mean. The other idea comes from the implementation of the Hausdorff metric on the CNN architecture. Cells contain time related voltage levels encoding the propagation time of a specific dynamic process. We used the maximum value to determine the Hausdorff distance but as a by-product each cell contains time related information.

If a system were able to generate and propagate trigger wave and measure the time required for the wave to reach a given position and store this information for each position then the associated map would contain all information to distinguish very sophisticated objects including also the previous examples. Let us define a gray-scale map, which is generated via wave

propagation in such a way where the gray-scale values are related to the time required for the wave to reach a given position. In this sense, this map contains dynamic information about differences between two objects.

For example in Figure 3.3 the wave map would be as it is shown in Figure 3.25. As we can see this map includes both the Hamming and nonlinear Hausdorff distances as special cases. It should be noted that while the Hausdorff distance can be determined at the end of the process only, the Hamming distance is already known at the start.

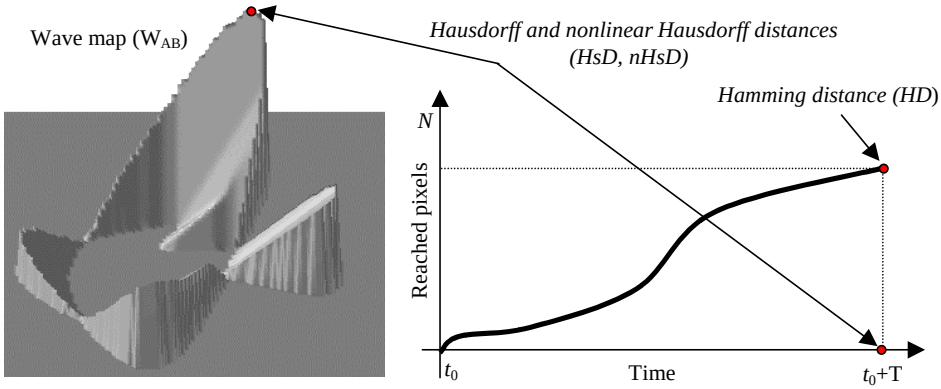


Figure 3.25 Wave map for objects in Figure 3.3. The plot of reached pixels by trigger wave versus time shows the properties of wave propagation. The number of pixels at end point gives the Hamming distance, while the time for reaching all pixels is equal to the (nonlinear) Hausdorff distance.

We can exploit that each position belonging to the Hamming distance contains a value related to the time required for the wave to reach that position. One of the possibilities is if these values (*local Hausdorff distances*) are summarized then both Hamming distance and Hausdorff distance are taken into account. Let us define a distance integrating the local Hausdorff distances over Hamming distance, assuming system continuous in space

$$WD(A,B) = \int_{U_{A\cap B}}^{U_{A\cup B}} W_{AB}(u)du \quad i.e.:$$
$$WD(A,B) = \int_{U_{A\cap B}}^{U_{A\cup B}} WD(u)du = \int_{U_{A\cap B}}^{U_{A\cup B}} HsD(u)\cdot HD(u)du = \int_{U_{A\cap B}}^{U_{A\cup B}} HsD(u)du\,; \qquad HD(u)\equiv 1 \tag{15}$$

where value of *HsD(u)* is intensity related to time of reaching *u(x,y)* position by trigger wave (*local Hausdorff distance*). The equation defines the Weighted Hamming (Integrated Hausdorff) distance based on the local Hausdorff distances. In discrete space the integration changes to summation. Two explanations can be given for the sum of the wave map.

- *Weighted Hamming*: The points of the Hamming distance are weighted where the intensity values of the wave map stands for these weights and these weighted Hamming points are summarized. Higher the intensity farther the distance of the point is from the initial position. It means that the farther points are more important than the near points.

- *Integrated Hausdorff*: The intensities of the wave map stand for the "local" Hausdorff distances. Local means that instead of the Hausdorff distance, in which the distance of the farthest point is computed, distances for each pixel are determined and stored (as if they would be the farthest points). Summing these values (in continuum it is the integration i.e. the volume of the wave map) gives the measure of the difference between two objects.

Both interpretations show that difference measurement relies on each point of the difference's set in which each point gives its specific value to the distance. Farther the point is from the intersection higher the weight with it contributes to the distance. It is opposite to the Hamming distance where each point has the same weight and it is also opposite to the Hausdorff distance where the distance of the farthest point results in the difference.

Theorem: The Weighted Hamming (Integrated Hausdorff) distance measure satisfies the metric axioms.[1]

Proof: The first and second axioms hold trivially.

(1) The $WD(A,B)=0$ iff $A=B$. $\rightarrow$ If $A\neq B$ then $WD(A,B)\neq 0$ therefore *HsD* will produce a nonzero value.

(2) $WD(A,B)=WD(B,A)$. $\rightarrow A\cap B$ and $A\cup B$ are the same in the two cases.

(3) Let's then verify the triangle inequality $WD(A,C)\leq WD(A,B)+WD(B,C)$. Hamming and Hausdorff distances are metrics. The Hausdorff distance is monotonically increasing on the area of Hamming distance: $HD_2 > HD_1 \rightarrow HsD_2 \geq HsD_1$. If we denote the Hausdorff distance on this area as a function, $f_{HsD}(HD)$ then we can write

$$
\begin{aligned}
WD(\mathrm{A},\mathrm{C}) &= \lim_{\mathrm{HD(A,C')}\longrightarrow\mathrm{HD(A,C)}} \int_{\mathrm{HD(A,C')}} f_{\mathrm{HsD}}(\mathrm{HD})dHD \\
&\leq \lim_{\mathrm{HD(A,B')}\longrightarrow\mathrm{HD(A,B)}} \int_{\mathrm{HD(A,B')}} f_{\mathrm{HsD}}(\mathrm{HD})dHD + \lim_{\mathrm{HD(B,C')}\longrightarrow\mathrm{HD(B,C)}} \int_{\mathrm{HD(B,C')}} f_{\mathrm{HsD}}(\mathrm{HD})dHD \\
&= WD(\mathrm{A},\mathrm{B}) + WD(\mathrm{B},\mathrm{C})
\end{aligned}
$$

The monotonically increasing property ensures that the $WD(A,C)\leq WD(A,B)+WD(B,C)$ holds.

Remark: From the proof it seems that if this metric is interpreted as a Weighted Hamming measure then any monotonic function over the Hamming distance can produce a metric. Of course the choice of the *weighting function* should reflect reasonable consideration.

For the presented example in Figure 3.21 the Weighted Hamming distances are monotonic for all the three cases, see Figure 3.39. It means that the Weighted Hamming can distinguish all

[1] It holds for initial set constructed as $U_{A\cap B\cap C}$.

the three cases. This shows the advantage of the Weighted Hamming distance calculation over the Hamming and the Hausdorff distances.

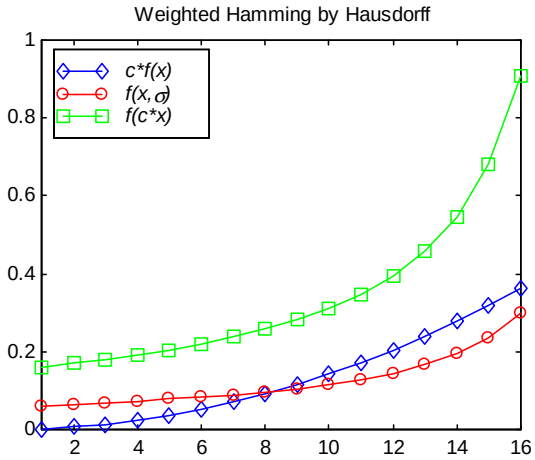


Figure 3.26 Weighted Hamming distance (WD) values for the three cases shown in Figure 3.21. In all the three cases the metric can distinguish objects.

3.4.2.1 Properties of the Weighted Hamming (Integrated Hausdorff) Metric

It comes from the definition of the Weighted Hamming distance that any difference that can be measured by the Hamming or the Hausdorff metric can also be measured by the Weighted Hamming metric. This fact relies on the monotonic connection that was used between the Hamming and the Hausdorff metric.

Additionally, the Weighted Hamming metric can solve such problems where the Hamming and the Hausdorff metrics are not satisfactory and produce false measurement.

The definition of the Weighted Hamming (Integrated Hausdorff) metric can be extended from the two-dimension space to n-dimension space quite naturally. That makes possible to measure difference between gray-scale objects as well. Another possibility is that this metric computation can be used not only in the field of image processing but any other type of applications where n-dimension objects should be compared. This investigation would go beyond of the scope of this work but the outline of the generalization is as follows. For instance, in image processing for gray scale objects the intensity values should be treated as an independent third “spatial” dimension next to the two regular spatial dimensions. Then the Hamming distance can generally be interpreted in such a way that any disagreement in the three dimensions (two spatial and one intensity) is encountered as difference. The Hausdorff distance measurement operates with these three-dimension sets to cover objects and change in extent of these covering sets gives the measurement of the difference. Therefore the Weighted Hamming (Integrated Hausdorff) uses 3+1 dimensions for wave mapping, three dimensions for “spatial” coordinates and the fourth is used for “intensities”. The volume of this four-dimension map produces the distance measurement.

Back to the 2D object comparison, some questions related to the metric properties will be investigated. For improved noise tolerance, the Weighted Hamming distance was restricted to the contiguous part of the intersection and the union of the sets A and B. Nevertheless, to ensure limitless treatment of object comparison, the definition of the Weighted Hamming distance

calculation should be extended to such cases where A and B objects do not have common part (i.e. the intersection is an empty set). Figure 3.27 shows possible situations for different cases. In general, it can be said if the intersection of object A and B is an empty set then the initial of the trigger wave should be the common weight point of objects A and B. This is a quite natural extension of the non-extreme case. In this case the propagation of the trigger wave is enabled over the whole system. The wave map integration takes place only on the area of the union of objects.

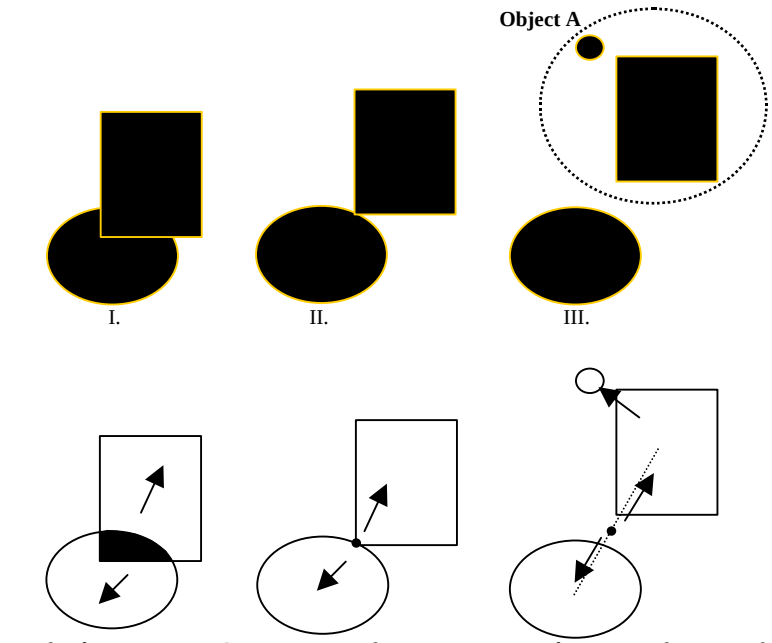


Figure 3.27 Extended Weighted Hamming distance interpretation for non contiguous intersection and union of objects. Wave map integration takes place only on the area of the objects.

Interesting question is the behavior of the Weighted Hamming distance to position errors, noise, scaling, etc. Because the lack of a regular treatment to characterize a given distance measure it seems to be a good way to use the term *sensitivity* from the field of electrical engineering and especially from the field of the electrical measurement and signal processing [41-44]. The *sensitivity* is defined as a measure of the extent to which a given circuit performance measure is affected by a given component within the circuit. The *sensitivity function* is a measure of the fractional change in some circuit characteristic, such as center frequency, to variations in circuit parameter, such as the value of a resistor. The sensitivity function is normally defined as the partial derivative of the desired circuit characteristic with respect to the element value and is usually evaluated at the nominal value of all elements.

So the *distance metric sensitivity* will be defined as the measure of the change in distance measurement effected by a given perturbation (noise, position, scale, …). Let us define the *relative distance metric sensitivity* - Q_D of a distance metric (D) to a given perturbation (h) as follows.

$$Q_D = \frac{\frac{\Delta D}{D_0}}{\frac{\Delta h}{h_0}} \approx \frac{h_0}{D_0} \cdot \frac{\partial D}{\partial h} \tag{16}$$

where D_0 and h_0 are values to normalize the sensitivity. The sensitivities of the Hamming, Hausdorff, and the Weighted hamming metrics are

$$Q_{HD} = \frac{h_0}{HD_0} \cdot \frac{\partial HD}{\partial h}, \qquad Q_{HsD} = \frac{h_0}{HsD_0} \cdot \frac{\partial HsD}{\partial h}, \qquad Q_{WD} = \frac{h_0}{WD_0} \cdot \frac{\partial WD}{\partial h} \tag{17}$$

In order to express the sensitivity of the Weighted Hamming metric with the other two sensitivities we write the partial derivative of the WD using the Equation (15) (definition of the Weighted Hamming metric) and apply the chain rule

$$\frac{\partial WD(A,B)}{\partial h} = \frac{\partial \int WD(u)du}{\partial u} \cdot \frac{\partial u}{\partial h} = HsD(u) \cdot \frac{\partial u}{\partial h} = HsD(u) \cdot \frac{\partial HD}{\partial h} = \int_{h_0}^{h} \frac{\partial HsD}{\partial h} dh \cdot \frac{\partial HD}{\partial h}, \qquad \frac{\partial u}{\partial h} \equiv \frac{\partial HD}{\partial h}$$

The partial derivatives of the Hamming and the Hausdorff metrics can be expressed via their sensitivities

$$\frac{\partial HD}{\partial h} = \frac{HD_0}{h_0} \cdot Q_{HD}, \qquad \frac{\partial HsD}{\partial h} = \frac{HsD_0}{h_0} \cdot Q_{HsD}$$

Finally we get that the sensitivity function of the Weighted Hamming metric is

$$Q_{WD}(h) = \frac{HD_0 \cdot HsD_0}{WD_0} \cdot \frac{1}{h_0} \cdot Q_{HD}(h) \cdot \int_{h_0}^{h} Q_{HsD}(h)dh \tag{18}$$

As we can see the $Q_{WD} \sim Q_{HD} \cdot Q_{HsD}^2$ in regard to the definition of the Weighted Hamming metric. This makes clear why the Weighted Hamming distance can solve untreatable problems for the Hamming or the Hausdorff distances. To demonstrate this property on examples different perturbation models were chosen. The distance sensitivity depends strongly on the chosen basic object as well. Here, compact objects were used, filled circles, rectangles and triangles. Figure 3.28 shows some objects and their possible perturbations. The unit of the perturbations was chosen as number of steps of the perturbation and h_0 is set to 1. The D_0 was chosen to the distance of the smallest perturbation. This means that it determines the minimal resolution of a given distance measurement respect to the minimal perturbation.

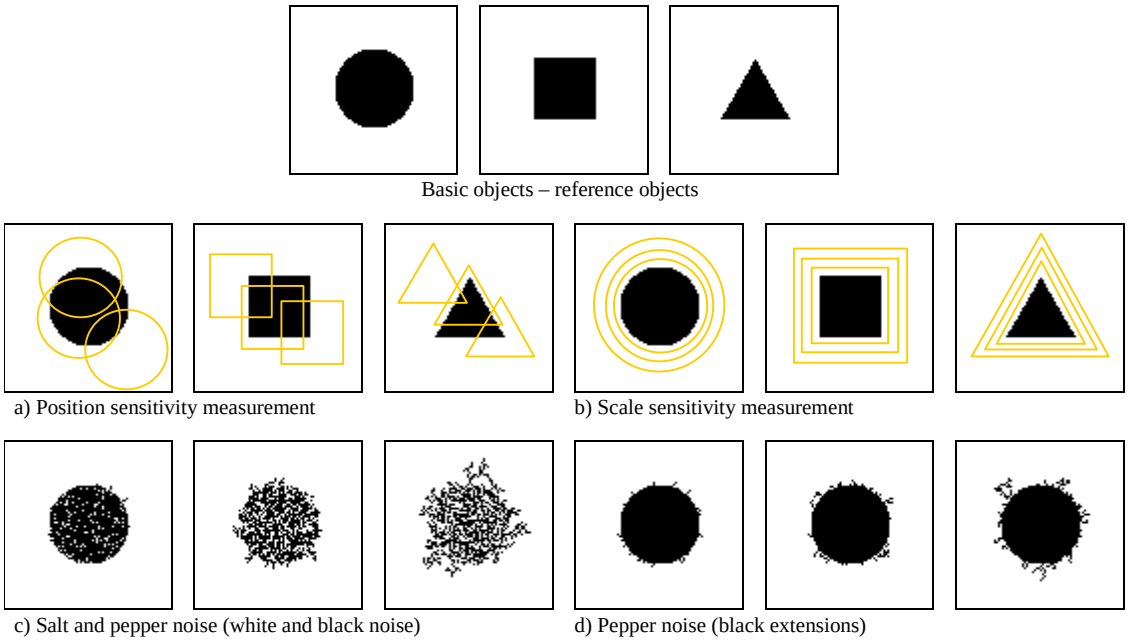


Figure 3.28 Basic shapes to test distance sensitivity using different types of perturbation. Disturbances can alter position, scale, rotation, and can cause different noise effects.

Next some measurements will be presented. The concrete values of sensitivity functions depend strongly on the choice of the basic object and direction. For instance, if the rectangular shape is sifted horizontally or vertically both the Hamming and the Hausdorff distance will be changed linearly. For this case, the normalized distances of metrics are shown in Figure 3.29. Therefore, the Weighted Hamming depends on the square of the shift. This is a specific case, different directions cause more complex values.

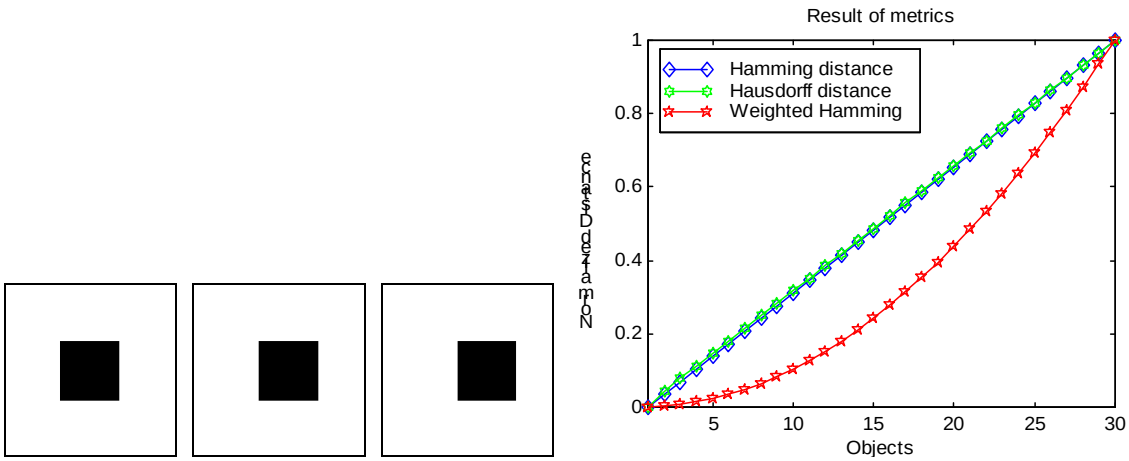


Figure 3.29 Metric values shifting the rectangular shape horizontally or vertically.

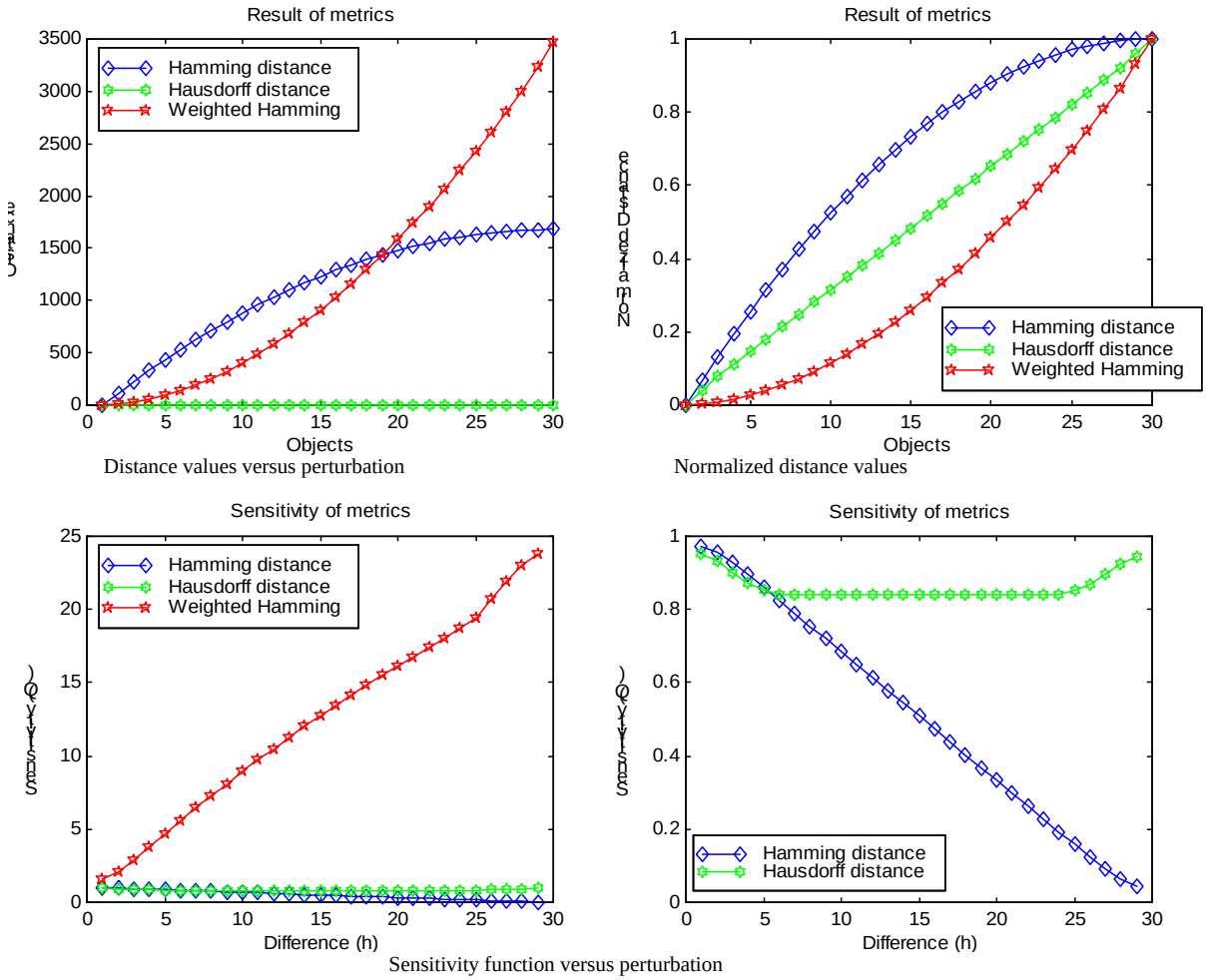


Figure 3.30 Sensitivity function of the Hamming, the Hausdorff, and the Weighted Hamming distances for position perturbation. Basic objects were shifted monotonically.

Figure 3.30 shows a typical result of metric measurement for position perturbation. The basic objects were shifted to a specific direction with given steps. The h_0 and D_0 belong to the first shift (one pixel shift). The direction of the shift affects the concrete values and their changes so the sensitivities of the metrics are different as well. But in each case the sensitivity of the Weighted Hamming was always strongly increasing.

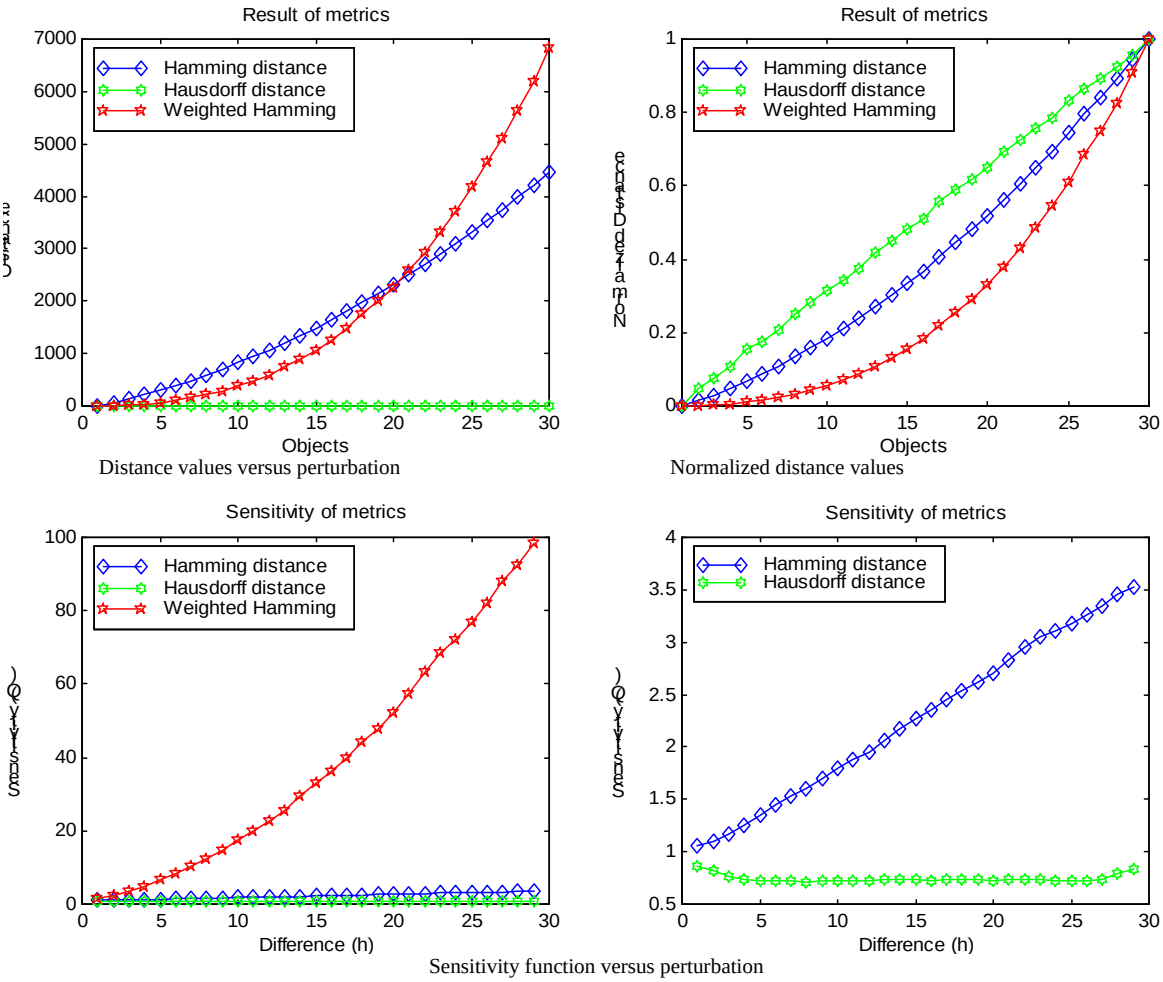
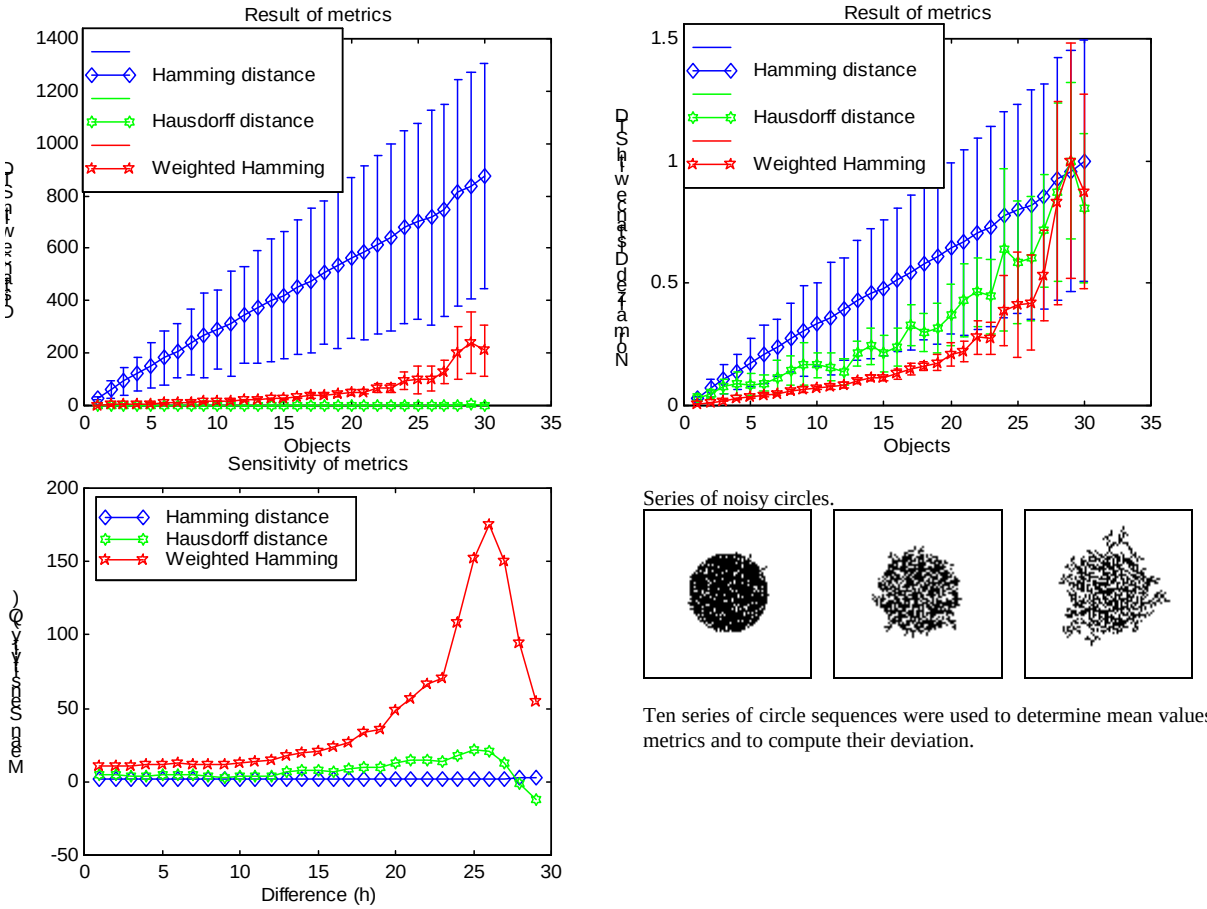


Figure 3.31 Sensitivity function of the Hamming, the Hausdorff, and the Weighted Hamming distances for scale perturbation. Basic objects were enlarged monotonically.

Figure 3.31 shows a typical result of metric measurement for scale (stretch) perturbation. The basic objects were enlarged monotonically respect to perturbation *h*. Here, the sensitivity of the Weighted Hamming is strongly increasing as well. It can be clearly seen that the Hausdorff distance is proportional linearly with the scale while the Hamming distance depends on the square of this scale (i.e. the area difference). So the Weighted Hamming distance is proportional with the third power of the scale.

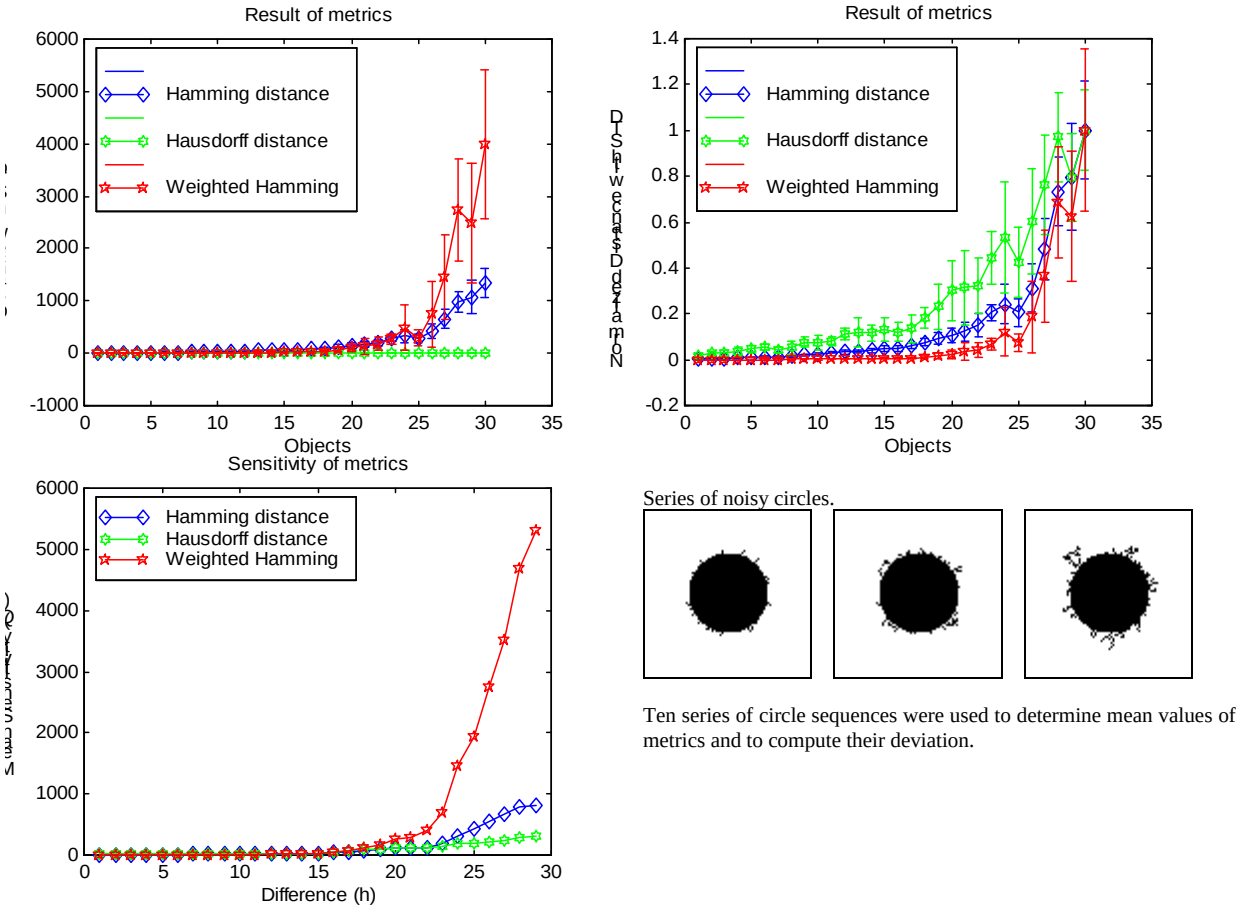
Third possibility to test sensitivity functions of metrics is to measure their change to a given noise perturbation. Among different types of pixel noise (gaussian, salt and pepper, …) the salt and pepper (impulse) and a special impulse noise were used (Figure 3.28/c and /d). In the first case, the probability of a white or a black spot was continuously increasing in the image series. In the second case, the probability of black extensions was increasing. Figures 3.32 and 3.33 show measurements for circle shape object. The other types of basic objects produce qualitatively similar results.



Ten series of circle sequences were used to determine mean values of metrics and to compute their deviation.

Figure 3.32 Sensitivity function of the Hamming, the Hausdorff, and the Weighted Hamming distances for salt and pepper noise (impulse) (Figure 3.28/c).

Figure 3.32 shows that Hamming distances have distinctively large deviation for impulse noise. At low noise level the Weighted Hamming has the smallest deviation. This indicates that noise circles are measured near to the basic circle shape. At high noise level the sensitivity of the Weighted Hamming increases remarkably. This non-linearity effect in sensitivity causes that Weighted Hamming has classification potential to determine whether the noisy shape is still circle or not. The Hausdorff distance has this potential as well but its volume is much less. The Hamming has constant sensitivity and therefore it can be used to determine the level of the impulse noise but not for classifying objects.



Ten series of circle sequences were used to determine mean values of metrics and to compute their deviation.

Figure 3.33 Sensitivity function of the Hamming, the Hausdorff, and the Weighted Hamming distances for noisy black extensions (Figure 3.28/d).

Figure 3.33 shows sensitivity values in case of noisy black extensions. Similarly to the previous case, the sensitivity of the Weighted Hamming has a strong change at a given noise level. This can be used to determine a threshold indicating that the noisy object has still a circle shape or not. The other two metrics do not have such a strong effect.

Another way to test metric sensitivities to measure distributions of metrics versus different distribution of perturbations. For instance, if the direction and the amount of the shift (or scale) depend on a stochastic variable then the distributions of metrics can be measured. This can model and qualify some types of optimization algorithms where model generation is necessary. The discrepancy of the generated model comparing to an ideal model and the possibility of a misclassification in a classification algorithm can be measured via distance sensitivity function.

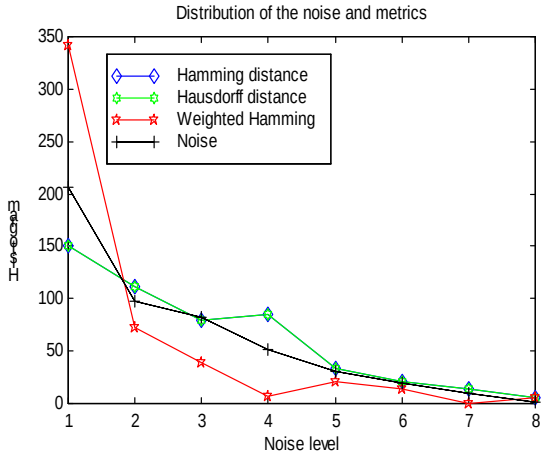


Figure 3.34 Metric distributions versus position depending on a Gaussian distribution. 500 samples were computed.

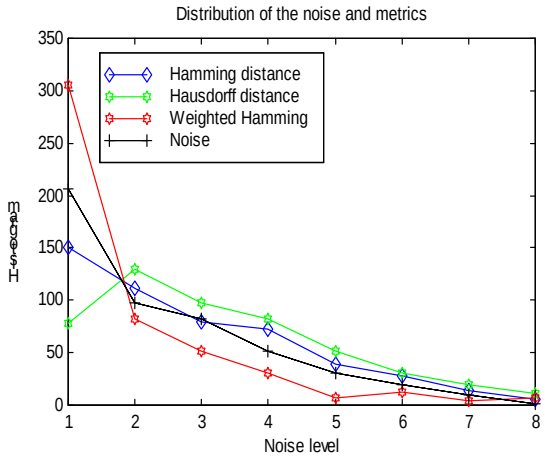


Figure 3.35 Metric distributions versus scale depending on a Gaussian distribution. 500 samples were computed.

Depending on the distribution of a stochastic variable metrics produce different distributions (Figure 3.34 and 3.35). Intuitively, we would expect that the distributions of the metrics were equal with the distribution of the stochastic variable. This would be the case if the sensitivities of the metrics depended on linearly with the disturbance. However, the sensitivities of the metrics determine how a given metric computation modifies the distribution, i.e. how the distribution is compressed or extended.

3.5 Weighted Hamming (Integrated Hausdorff) Metric Computation on CNN Architecture

Until now, theoretic considerations led us to a novel type of metric, which is more robust to measure object's differences. Important question is how to calculate efficiently this metric. The determination of the Weighted Hamming metric requires huge computational power independently of the actual method. For each position the local Hausdorff distance should be calculated and finally summarized. As we have seen the wave approach is a good candidate to solve this task because via a wave transient the wave fronts explore each position and their running time encodes those local Hausdorff distances. The CNN architecture provides an ideal

tool to generate wave processes therefore the wave map computation can be solved very fast. Next the implementation of the Weighted Hamming distance on the CNN architecture will be considered.

3.5.1 Implementing the Weighted Hamming (Integrated Hausdorff) Distance on CNN

Two types of implementation will be discussed. One of them is a straightforward method called dynamic solution the another one is an iterative solution. The VLSI complexity of the first method is sophisticated, and current CNN chips do not support, nevertheless this one takes all advantages of the CNN dynamics. The second one is more sophisticated from algorithmic point of view but requires only binary morphology operations implemented with linear templates on CNN and can be tested on present CNN chips.

3.5.1.1 Dynamic Solution

Below we discuss the dynamic solution in details, i.e. how the wave propagation can be recorded and summarized. Figure 3.36 shows a possible implementation of the Weighted Hamming distance computation on two-layer CNN. The advantage of this implementation is that several object-model pairs can be compared at the same time.

The idea is the same with the implementation of the Hausdorff metric on CNN shown in Figure 3.15. The difference is that we exploit the sub-result of the process, namely, the generated Wave Map. This Wave Map contains in each position a voltage level that encodes the time required for the trigger wave to reach the given position. This agrees with the local Hausdorff distance (*HsD(i,j)*) and its summation gives the Weighted Hamming distance. Applying average and threshold operation in sequence, the computed value might be used in a classification decision and at last those objects might be recalled where these differences are large indicating large difference.

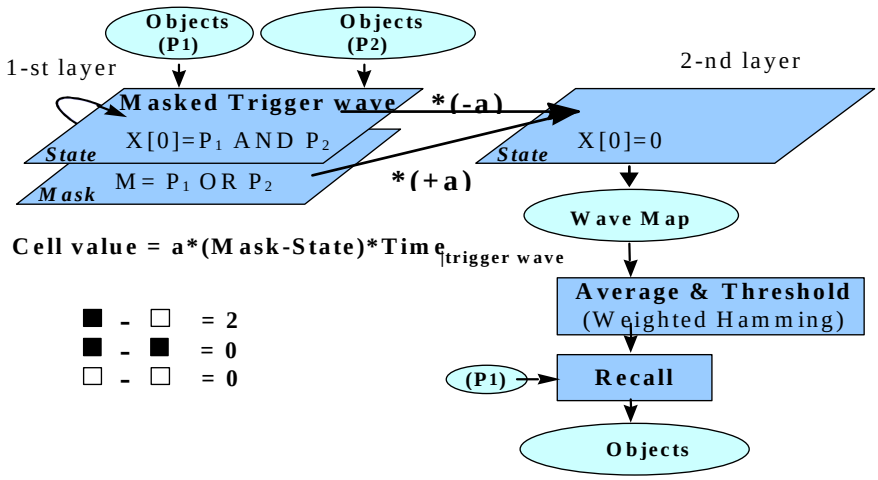


Figure 3.36 Dynamic implementation of Weighted Hamming (Integrated Hausdorff) distance measure on CNN architecture. From the intersections of sets to be compared trigger waves propagate on the first layer and time is measured via constant current filling on the second layer. The current term has only three possible combinations. At the end of the process each pixel contains time related voltage level and these values are summarized. The computed distance can take place in a classification process.

3.5.1.2 Iterative Solution

The iterative method for wave map generation is presented in Figure 3.37. Its advantage is that it can be tested on present CNN chips although it is a time consuming algorithm. The number of iterative steps depends on object's sizes to be compared. This is also similar to the iterative type implementation of Hausdorff metric on CNN presented in Figure 3.16 except that the Wave Map is summarized.

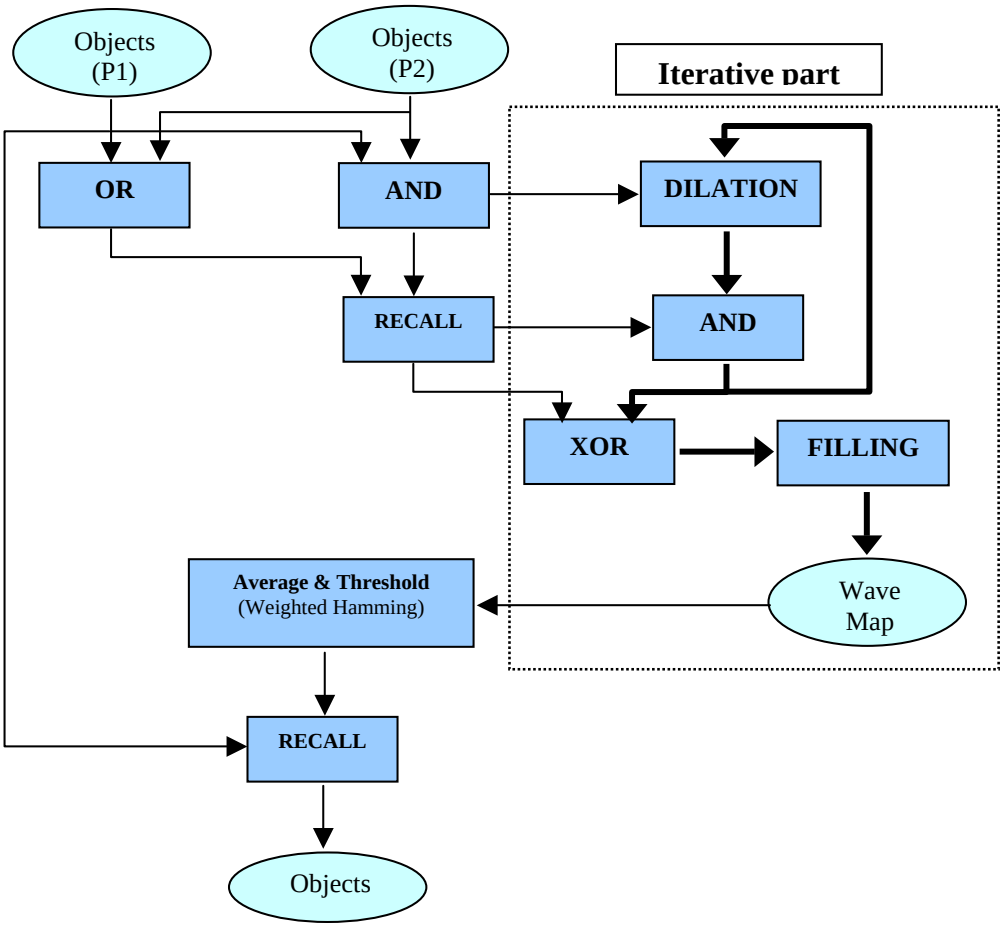


Figure 3.37 Iterative implementation of Weighted Hamming measure on CNN. From the intersections of sets to be compared a binary morphology operation namely dilation expands the objects width one pixel in each step. The result of a step is used to fill a layer with a constant current for a given time producing quantized map. At the end of the process the complete wave map is obtained. This wave map is summarized and used as a distance for classification.

3.5.1.3 Hardware Requirements of Dynamic and Iterative Solutions

We have described two possible implementations of Weighted Hamming (Integrated Hausdorff) distance computation on CNN architecture for object segmentation and classification. The discussed dynamic solution requires the so-called fixed-state map technique, nonlinear cell interactions, and it is a two-layer approach. The VLSI implementation complexity of the solution mainly depends on the implementation of wave generation, since this is the only building block which require nonlinear template interaction.

3.5.2 Implementation result of the Weighted Hamming measure on the 64x64 CNN-UM chip (ACE4K)

In 1999 an operational 64x64 sized (4096 analog processor elements on a chip) programmable CNN Universal chip (called cP4000) [11-12] with analog, binary and optical inputs and analog and binary output CMOS chip was designed in Seville. The test results shows that the chip is operational, and it fulfills the accuracy requirements. This means that the cP4000 chip is a breakthrough of the CNN technology which opens the gates to industrial and commercial applications. A computational framework [102, 107, 108] was built around the new 64×64 CNN Universal Chip designed in Seville and among many possible applications the implementation of the wave metric was managed to be solved.

The iterative method of Weighted Hamming distance (shown in Figure 3.37) was implemented on the 64x64 I/O CNN-UM chip. This method requires binary morphology operations [28], fixed state map techniques, and linear templates. Figure 3.38 shows an example where this wave map generation on ACE4K chip is presented.

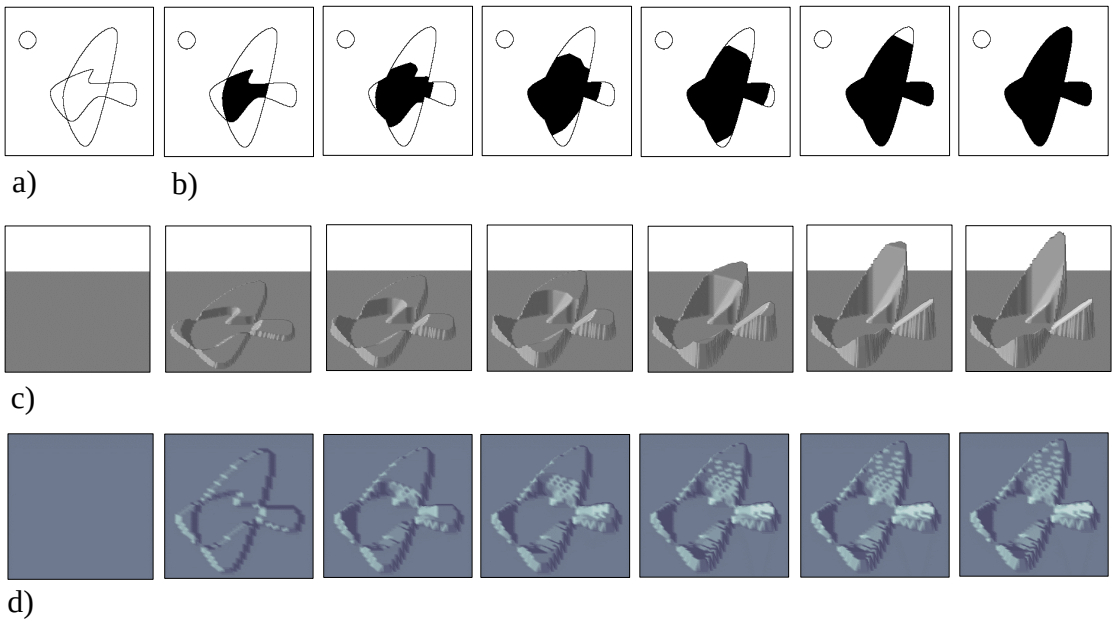


Figure 3.38 Wave Map generation. a) Outlines of two partially overlapping point set, b) Trigger wave spreads from the intersection through the union of contiguous part of point sets until all the points become triggered, c) Wave map generated by increasing intensities of pixels until trigger wave reaches them, simulation result d) Consecutive steps of generating Wave Map on the 64x64 I/O CNN-UM chip (ACE4K).

Note: the intermediate steps are shown for demonstration purpose only.

For the presented example in Figure 3.21 the Weighted Hamming distance which was investigated as a more usable distance calculation than other metrics produces monotonic function for all the three cases both in simulation and on chip (Figure 3.39).

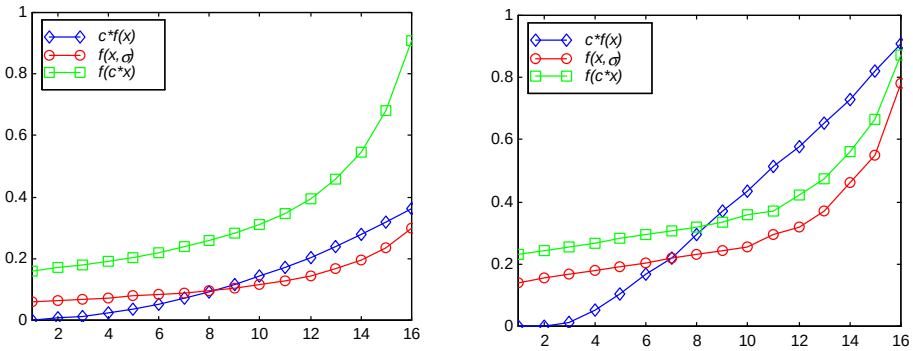


Figure 3.39 Weighted Hamming values for the three cases shown in Figure 3.3. a) simulation results, b) results of the iterative implementation of Weighted Hamming distance calculation on the 64x64 I/O CNN-UM chip made in Seville (ACE4K).

Figure 3.40 shows simulation and experimental results computing different distance measures between objects shown in Figure 3.21.

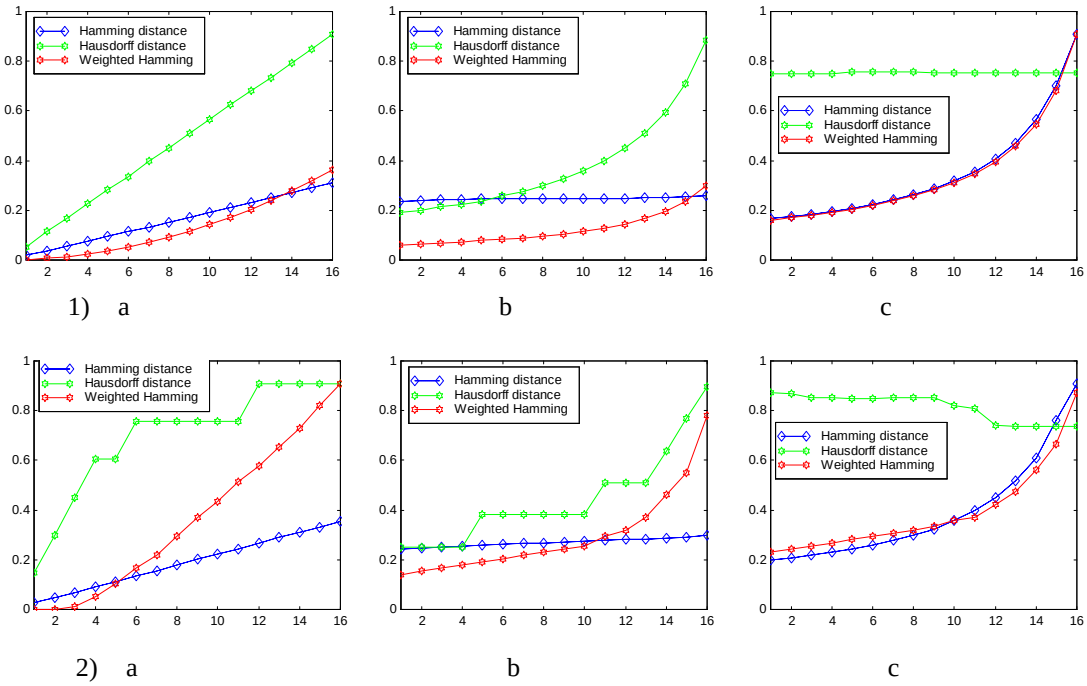


Figure 3.40 Results of three distance measurements between objects shown in Figure 3.21. The distance methods are Hamming, Hausdorff, and Weighted Hamming (Integrated hausdorff). 1) simulation results for the three cases, 2) measurement on the 64x64 I/O CNN-UM chip.

Comparing the measurements of Hausdorff distances to the simulation their steps-like functions might seem as disadvantage. The reason is that the test images cover such a broad "spectrum" that the speed of trigger wave has a strong dependence on the object's width. Therefore for each object pair the propagation speed of trigger wave should have been tuned to achieve constant propagation speed. Although the Hausdorff distance is involved at the weighted

Hamming distance calculation this dependence is eliminated. This shows the robustness of the proposed distance metric.

3.6 Generalized Methodology for 2D Object Shape Comparison and Distance Computation

As we have seen in the previous sections, the choice of the distance calculation bears important effect upon separation and differentiation among objects. A novel type of class of comparison might be the method if geometrical (structural and morphology) properties of shapes are extracted by spatio-temporal dynamical processes. The time evolution of these nonlinear dynamical processes carry additional information about objects and this "hidden" information serves as a key to distinguish objects which are untreatable by other metrics. The new approach is here that for comparison of 2D objects trigger waves make a survey of objects and the time evolution of this propagation is recorded and transformed into a spatial map. Several metrics can be obtained from this extracted information. These metrics measure both differences and similarities among objects. They include dynamic type information of properties of objects not only static ones. Static properties are, for instance, the sum of points or a distance of a given point. Dynamic information, on the other hand, carries data related to properties of dynamic processes. In this sense the rich world of dynamical phenomena are used to explore objects.

3.6.1 Object Comparison Based on Spatio-temporal Dynamic Transformation

For the possibility of a general discussion some notations and formalisms are presented here. Data to be processed can be binary, real valued images or real valued functions.

Notation and Definition

$U_A : U_{A,MxN} \rightarrow \{-1,+1\} \in \Im^2$ binary images

$V_A : V_{A,MxN} \rightarrow [-1,+1] \in \Re^2$ gray-scale images

$v_A : v_{A,1xK} \rightarrow [-1,+1] \in \Re$ real value function

General Theory

The keys of the classification process based on wave metric are wave based transformation of objects to be compared, intermediate processing of wave mapping, and distance calculation which itself can solve the classification (shown in Figure 3.41). The method resembles the transformation based comparison where usually images are projected into a different space. Here, the transformation is based on wave mapping and space remains the 2D image space. The intermediate processing is required for further data compression and finally, distance calculation which can be involved in the previous step determines the degree of coincidence of objects. If steps of wave map based calculation are chosen carefully then the final distance value can be used for also classification.

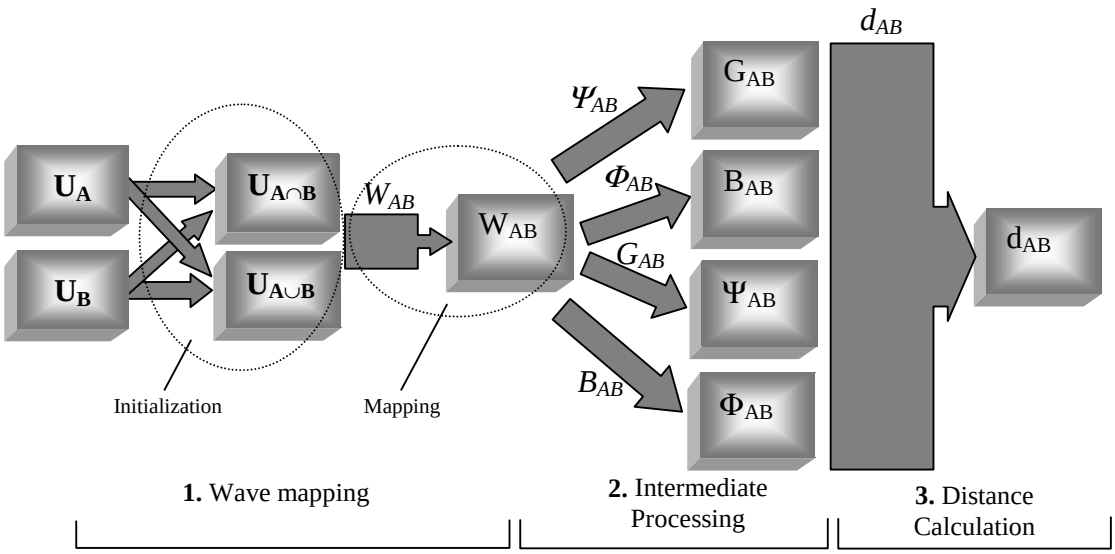


Figure 3.41 General form of wave map based distance calculation for object comparison. Three major steps are wave mapping, intermediate processing, and finally distance calculation. Wave mapping generates from the initial sets a spatial map in which time dependent information is encoded. This information is further processed and finally, distance calculation determines in a single number the degree of coincidence of objects.

1. Wave mapping

Wave mapping contains two steps, namely

- Initialization – construct sets where dynamic mapping takes place.
- Mapping – a dynamic process explores the space of sets.

Up to now we applied the following projection:

Projection type: $W:\Re^2 \to \Re^2$

$$\mathbb{W}_{AB}: \mathrm{V}_{AB} = W(\mathrm{U}_{A\cap B}, \mathrm{U}_{A\cup B})\big|_{\text{except incontiguous parts}}$$

Wave processing results in a gray-scale image. This wave map contains the time evolution and dynamics of a propagation type process as an intensity map in such a way that the intensity value at a given position increases continuously until the wavefront reaches that position. This wave map carries all information that can be used for many different distance and comparison measurements. From this wave map several measures can be computed depending on the chosen aspect, i.e. area integration, boundary scan, histogram analysis, skeleton based summarization, etc. The optimal choice depends on the type of the problem.

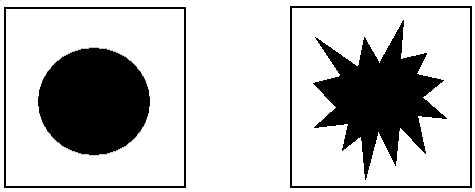
One subclass of this method was presented in the previous section as Weighted Hamming metric including the Hamming and Hausdorff distances as special cases.

The wave map makes possible to encode the similarity of the objects in one number instead of computing several features that would require many different computations. In classification tasks it might be useful to make decisions based on one number instead of a feature vector.

In the projection the initial sets were up to now the union and intersection of objects. Question may arise what is the reason behind this. Generally it is possible to use other initials but in the most cases this seems to be an optimal choice. But on the other hand, if the task is to

measure boundary roughness or deformation of objects then other initials can be used (see Figure 3.42). In this case the II and III method is more suitable for a measurement because the significant information is extracted out, namely, the intensity values at the boundary of objects are examined.

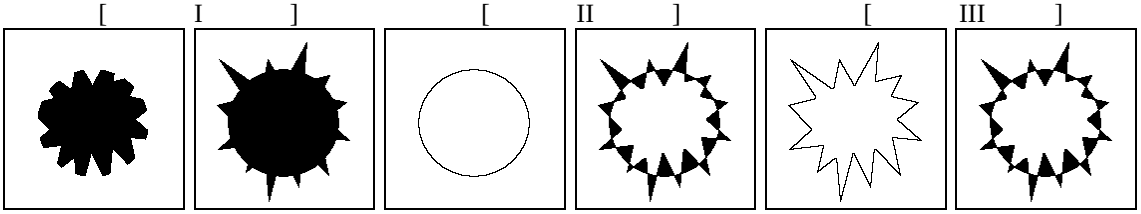
Objects to be compared. A is the reference object and B is the object what it should be characterized from deformation point of view.



A B

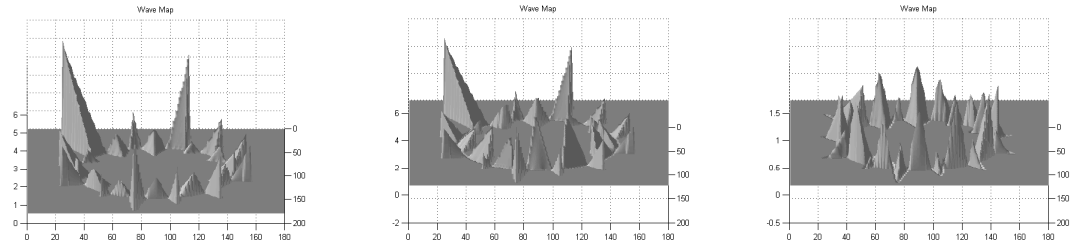
Possible initials for wave mapping. Three different cases are presented (I, II, III). In each case the first image is the starting set (initials for dynamic process) while the second is the final set (where dynamic process may take place). The function $b(.)$ means boundary of object.

[I] [II] [III]



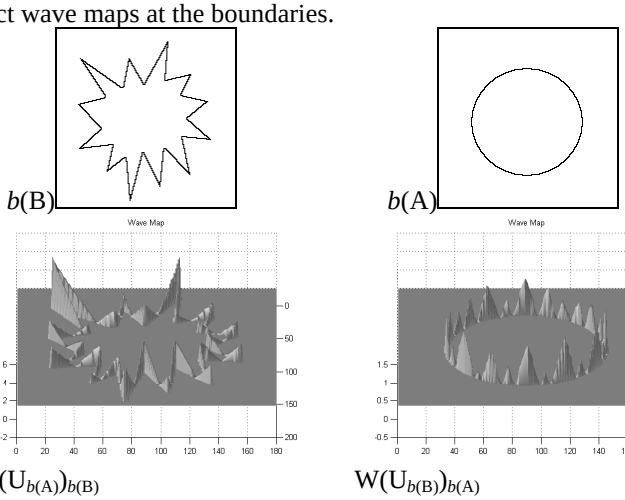
$A \cap B$ $A \cup B$ $b(A)$ $b(A) \cup b(B) \cup A \otimes B$ $b(B)$ $b(A) \cup b(B) \cup A \otimes B$

Wave maps for I II III



$W(U_{A \cap B}, U_{A \cup B})$ $W(U_{b(A)}, U_{b(A) \cup b(B) \cup A \otimes B})$ $W(U_{b(B)}, U_{b(A) \cup b(B) \cup A \otimes B})$

In case of II and III it is better to restrict wave maps at the boundaries.



$b(B)$ $b(A)$

$W(U_{b(A)})_{b(B)}$ $W(U_{b(B)})_{b(A)}$

Figure 3.42 Wave mapping initials for surface roughness measurement. Different initials result in different mappings and finally different distances. Nevertheless distances will be qualitatively equivalent. To avoid redundant information cases II and III restrict wave map values only at the boundaries of object in order to contain the most relevant information.

Although the case I contains much more information but this seems to be redundant from point of view of roughness measurement. Therefore in cases II and III wave maps are restricted to a specific boundary. If we focus on the rough object B then case II can be interpreted as each point of the boundary of rough object B contains its distance to the boundary of reference object A. Here, instead of case I, the numbers of deformation is connected to the object B. In case III, the focus is put on the reference object A. The deformation is projected to the boundary of the reference object, therefore "without knowing" about the rough object the deformation can be related to the reference. To find out what the boundary scan means, without loss of generality, it is enough to examine the small part of boundaries, in which the boundary of reference object looks like a straight line. Figure 3.43 shows an example.

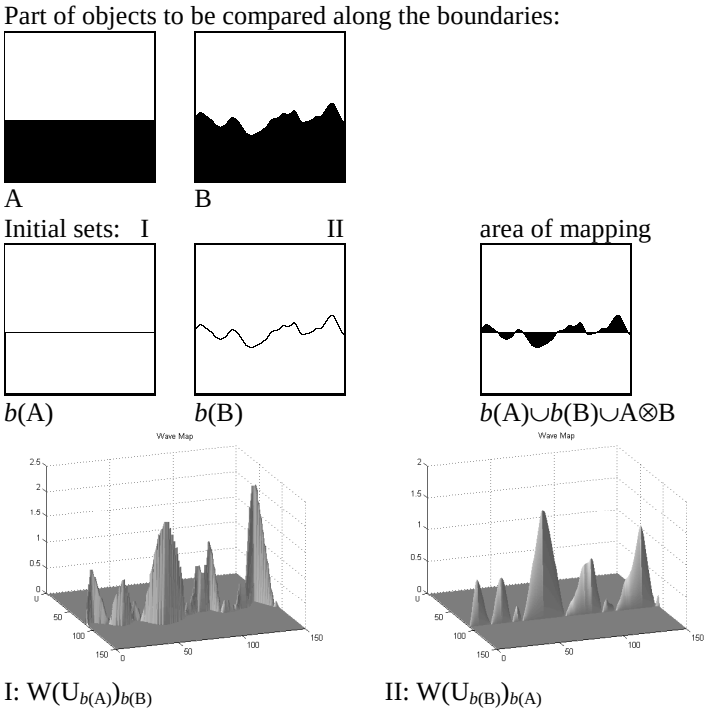


Figure 3.43 Roughness measurement restricting wave maps to boundaries.

If we analyze restricted wave maps it turns out that the Weighted Hamming measure on boundary in case I will give back the Hamming distance. Intensity values at boundary are the values of local Hausdorff distances and it is quite clear, that they are proportional to the vertical segments of the Hamming distance. Therefore its integral gives back the Hamming distance.

$$\begin{aligned} &\text{case I}: d_{WH}(U_{b(A)}, U_{b(A)\cup b(B)\cup A\otimes B}) = \int W(U_{b(A)}, U_{b(A)\cup b(B)\cup A\otimes B})_{b(B)}\, dxdy, \\ &d_{WH}(U_{b(A)}, U_{b(A)\cup b(B)\cup A\otimes B}) = » \cdot d_H(U_A, U_B) \end{aligned} \tag{19}$$

The second case in Figure 3.43 contains also similarly the Hamming distance, although the propagation alters the final result. Its advantage is that the roughness values are projected to a straight line, therefore the analysis much easier. For visualization purposes it is quite convenient to draw it as a one dimensional function, where function values related to the boundary of the reference object show the difference of the rough object.

2. Intermediate processing

The previous investigations lead us to the next steps of the wave based distance calculation. In the intermediate processing the wave map is modified in order to filter out the relevant information. This is different then in the first step of the wave mapping. There, the initial sets were analyzed, while here the wave map is processed. Such a process can also be the boundary restriction of the wave map. Another possibility is, for instance, if the time evolution of the wave map is projected into an one dimensional function. Generally, transformations can be grouped into four classes depending on their projection.

Time distribution calculation.

Projection type: $\ddot{} : \Re^2 \to \Re$

$\ddot{}_{AB} : v_{AB} = ?(V_{AB})$

Possible operations: min(.), max(.), mean(.), median(.). etc.

Spatial distribution (histogram) calculation.

Projection type: $F : \Re^2 \to \Re$

$F_{AB} : v_{AB} = F(V_{AB})$

Possible operations: min(.), max(.), mean(.), median(.), mode, etc.

Gray-scale morphology

Projection type: $G : \Re^2 \to \Re^2$

$G_{AB} : V_{AB} = G(V_{AB})$

Possible operations: min(.), max(.), mean(.), median(.), etc.

Binary morphology

Projection type: $B : \Re^2 \to \Im^2$

$B_{AB} : U_{AB} = B(V_{AB})$

Possible operations: edge, corner detection, etc.

3. Distance calculation

Projection type: $d : \Re \to \Re$

$d_{AB} : v_{AB} = d(v_{AB}, u_{AB})$

Possible operations: min(.), max(.), mean(.), median(.), mode, etc.

The nonlinear wave mapping based metric computation can be specified by three steps as follows.

$$D = [W, \overset{I.}{\{?, F, G, B\}}, \overset{II.}{d}, \overset{III.}{d}] \tag{20}$$

The three parts corresponds to a three-step transformation where projections compress spatio-temporal information by reducing spatial and temporal dimensions into a single real number that gives finally the result of the metric computation. The D metric or distance has the steps as follows.

I. The *W(.)* is a spatio-temporal mapping generating a wave map.
II. The $\{?, F, G, B\}$ intermediate processing computes a specific distribution or gray-scale or binary morphology computation.
III. The d identifies the final distance via a real function projection.

Discussion

Properties and connection between wave mapping, time distribution, and histogram calculation are discussed.

1. Connections between distribution and histogram calculation

As we have seen in Example A:

Wave processing: $W_{AB} = W(U_{A\cap B}, t)\Big|_{U_{A\cup B}}$ except incontiguous parts

If otherwise is not stated, initial sets are the intersection and the union of objects except incontiguous parts. We introduce for intermediate processing two functions projecting wave map to space distribution and time distribution.

Time distribution of the wave map: $\ddot{\ }_{AB} \hat{=} \ddot{\ }(W_{AB}, t)_{|U_{AB}}$

This denotes for the distribution of the wave map related to the time. It means that at each time *t* the function $\psi_{AB}(t)$ has the area of reached region by the wave process, i.e. the number of reached positions in discrete space.

Spatial distribution of the wave map: $\Phi_{AB} \hat{=} H(W_{AB}, u)_{|U_{A\cup B}} \equiv F(W_{AB}, t)_{|U_{A\cup B}}$

This denotes for the histogram of the wave map. Because during the wave process an intensity value encodes the corresponding propagation time, this definition equals if the distribution is expressed versus time (see Figure 3.44).

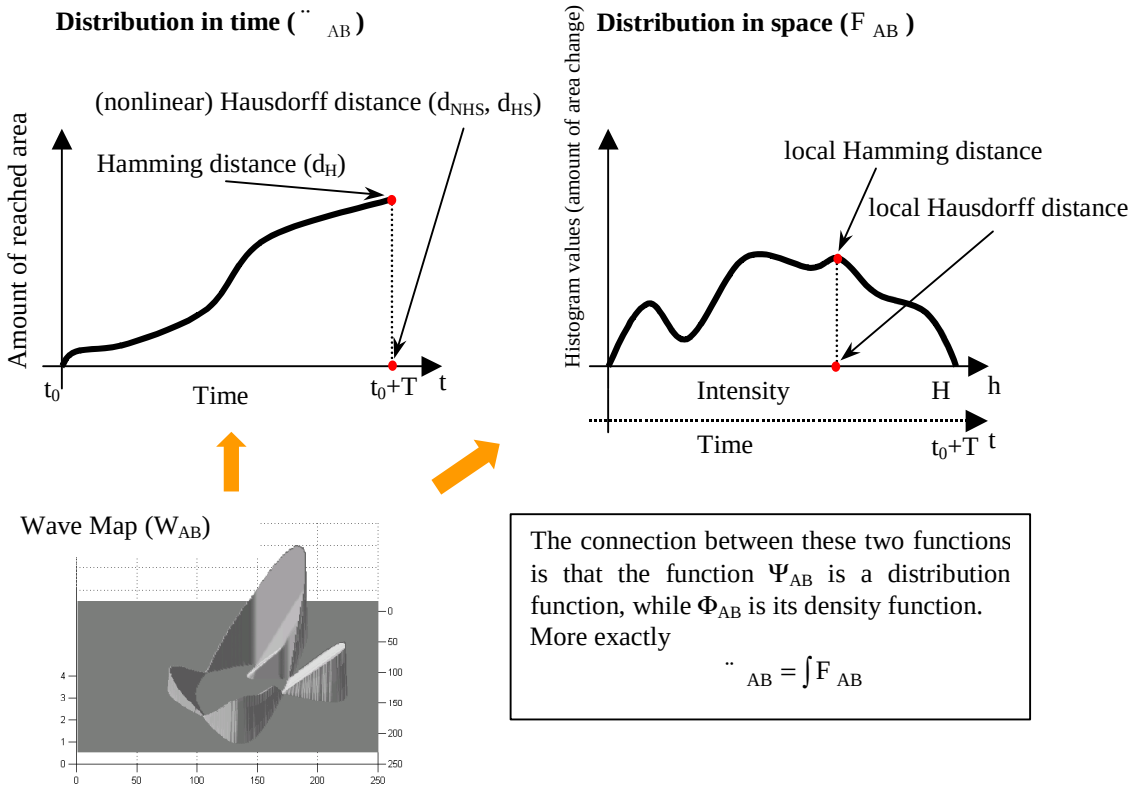


Figure 3.44 Projections of wave map to time distribution and to space distribution.

Different distance calculations can be expressed from the distribution functions.

$$\text{Hamming distance}: \text{HD} = \int_0^H \text{F}_{\text{AB}}(\text{h})dh \equiv \frac{1}{T}\cdot\int_{t_0}^{t_0+T}\text{F}_{\text{AB}}(\text{t})dt$$

$$\text{Hausdorff distance}: \text{HsD} = \text{H} \equiv \text{T}$$

$$\text{Weighted Hamming distance}: \text{WD} = \int_0^H \text{h}\cdot\text{F}_{\text{AB}}(\text{h})dh \equiv \int_{t_0}^{t_0+T}\text{t}\cdot\text{F}_{\text{AB}}(\text{t})dt$$

2. Connections between time and spatial distribution

It will be shown that the Weighted Hamming distance via local Hausdorff distance can be computed already from the intermediate process using distribution calculation. This is important from implementation point of view.

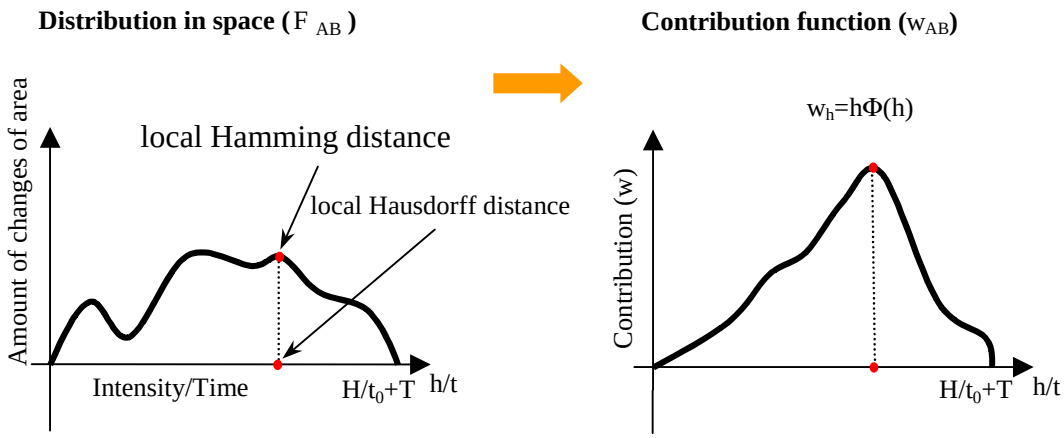
$$WD = \int_{t_0}^{t_0+T}\text{t}\cdot\Phi_{\text{AB}}(\text{t})dt = \int_{\text{U}_{\text{A}\cup\text{B}}}\text{W}_{\text{AB}}(\text{U}_{\text{A}\cap\text{B}})du$$

For instance, if the iterative type implementation of the wave map generation is used then in each time step the distribution functions can be computed very simply using the first type of equation. The area integration belongs to the dynamic type implementation.

An other useful function can be which also is computable during the wave map generation is the

Contribution function of the wave map: $w_h(h) = h \cdot F_{AB}(h)$
$w_t(t) = t \cdot F_{AB}(t)$

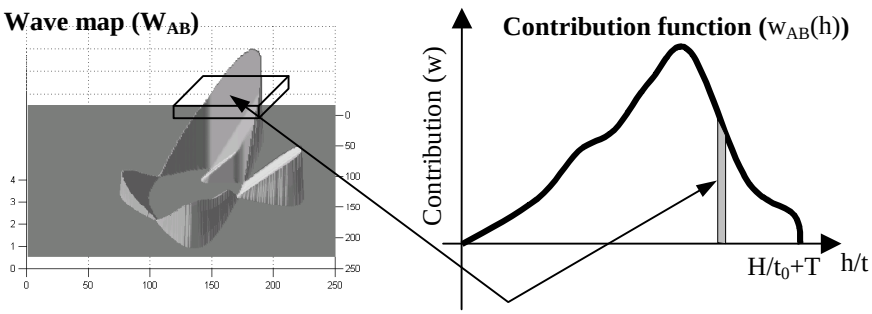
It denotes for the local weight contribution, i.e. how many pixels belong to a given Hausdorff distance (multiplication of the local Hamming distance and local Hausdorff distance).



Using contribution function the Weighted Hamming can be expressed as:

$$WD = \int_0^H w_h(h)dh = \int_{t_0}^{t_0+T} w_t(t)dt$$

The connection between wave map and contribution function can be seen in the following picture.



Weighted Hamming distance: $WD = \int_{U_{A\cap B}} W_{AB}(U_{A\cap B})du = \int_0^H w_h(h)dh$

When might be useful to use this contribution function? If we would like to get additional information about the distribution of the wave map then plot of this function shows it in a very impressible way. Figures 3.45-46 show different distribution functions of simple geometrical patterns.

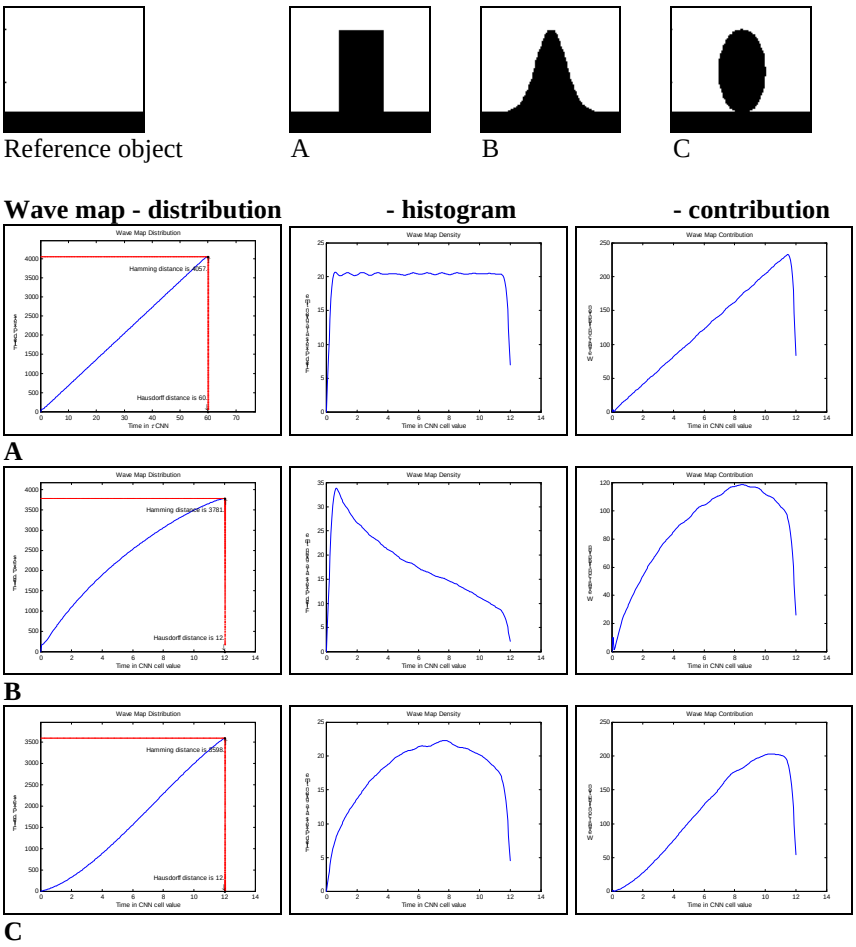


Figure 3.45 Different distribution type functions of the wave map for simple geometrical patterns. Functions are distribution, histogram, and the contribution function.

As it can be seen useful properties can be extracted from the distribution functions. The distribution function gives a general characterization of the difference. Its derivative function, the histogram shows especially what region of the difference gives the largest change while the contribution function determines what difference results in the largest contribution to the distance. Applying operators like maximum, mean, median, etc. to these last two functions other distance measures can also be defined.

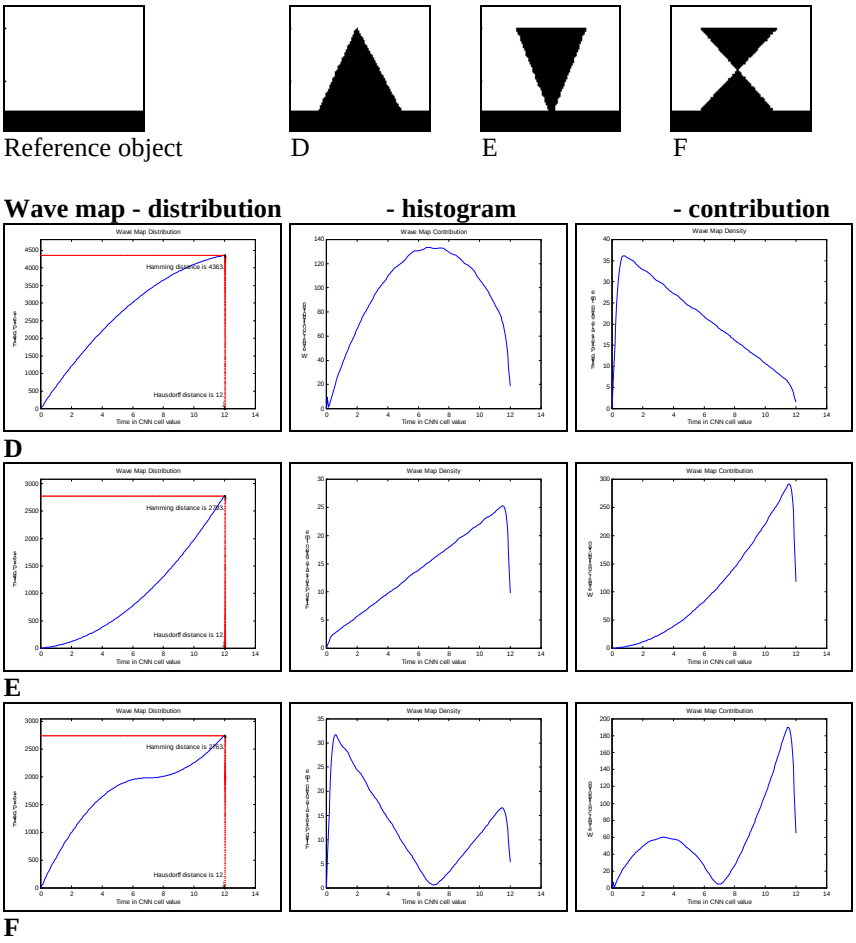


Figure 3.46 Different distribution type functions of the wave map for simple geometrical patterns. Functions are distribution, histogram, and the contribution function.

3.6.2 Static versus dynamic type of comparison

The differentiation among metrics based on dynamical aspects is of course heuristic, nevertheless relies on a very important matter. All the metrics which exploits possibilities of spatio-temporal dynamical processes for object shape discovery contains the potentiality to code time related information. This type of information as a new dimension additionally to the spatial dimensions makes possible to extend features of objects that are hidden for a spatial investigation. This produces such a class of metrics or distance measures that they can solve untreatable problems for other types of metrics. The main advantage is that information can be compressed either into a single number instead of a vector of features nevertheless being able to separate difficult cases. Another possibility is to address not only the distance measure itself but also the problem of similarity measure. Here, it was not our goal to investigate this area but it seems to be quite intriguing and requires systematic and thorough research.

Question might arise whether the generalized methodology based on spatio-temporal processes could be brought into any class of previous types of metrics. Two types of widely used classes of metrics might come to the fore. The two ways are the image metric and the transformation based metric, respectively. The two concepts are interpreted as follows.

Image metric: any measurement takes place in the image plane, e.g. computing Euclidean distances between corresponding pixels in the two images.

Transformation based metric: the measurement takes place in different space than image plane. The pixels or properties of objects are transformed into a space in which the operation of this transformation can be coded in a compact form (e.g. affine transformations are represented with their transformation matrices). The differences between objects can be calculated as the differences among transformations (e.g. differences among matrices).

Thence it arises the problem again that objects might differ from each other in many ways. A possible case is shown in Figure 3.47 where objects are closer or further depending on the chosen metric.

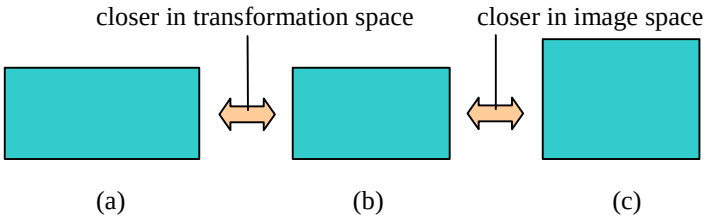


Figure 3.47 The object (b) might be closer to the object (a) when the difference is computed in transformation space, and closer to the object (c) when the difference is computed in image space.

The introduced spatio-temporal process based metric is a transformation metric on the one part but it is on the other hand an image metric as well. The spatio-temporal process transforms objects and/or their properties via extracting dynamical information but this extracted information remains in image space, i.e. a gray-scale image is generated. The additional operators working on this gray-scale image are image operators as well. Therefore this metric might embody the advantages both of image metrics and transformation metrics.

3.6.3 Implementation in a CNN Universal Chip environment

The idea to use spatio-temporal dynamical processes to explore object shapes would not be feasible if we do not have a massive parallel architecture to process it. All the results here would be only theoretical and the hope to use it in real applications would be an illusory. The CNN Universal chip environment provides a general framework to implement wave-based metrics efficiently. In the previous sections necessary subroutines for the CNN implementation were already discussed. Wave propagation, wave map generation is solved and its implementation on the 64x64 sized chip was presented. There the iterative type of implementation gives a second possibility, i.e. to compute different distribution functions of the wave map. Thereby other useful information can be extracted at the same time of wave map generation and more precise analysis can be obtained. The major advantage of the dynamical implementation is that the time requirement is the range of the spatio-temporal dynamical processes and their fast transients makes possible to use it in sophisticated real time applications.

3.7 Conclusions

I have introduced and analyzed a novel type of metrics for 2D object comparison. These metrics exploit the possibilities of spatio-temporal dynamical processes. Geometrical properties of objects are extracted via different types of propagating waves and time evolution of this propagation encodes the information about similarity and difference. This information contains dynamical type features compared to previous studies in which the extracted properties are rather static. Examples show many advantages of these metrics and demonstrate that they can be used efficiently for object classification. Especially if time constraints do not make possible to extract many features of objects but they would be needed for a robust testing then a complex attribute value can help.

Previous studies show that difference measurement cannot be really treated without complex aspects of similarity. Broader environment of a problem affects strongly that a given distance measurement how much is efficient in distinguishing of objects. This work argued strongly to use spatio-temporal dynamical processes in difference measurement and their flexibility make possible to address similarity as well. Hopefully the rapid grow of CNN technology provides a flexible tool to compute these methods efficiently.

The dynamical implementation of the Hausdorff metric via CNN makes possible to use this metric in problems in which real-time processing is required and all this can be packaged into a compact hardware without using huge work-stations.

The Weighted Hamming metric seems to be also an efficient method for object shape comparison. Its generalization to n-dimensional space makes possible to compare not only 2D binary objects but gray-scale objects as well. Additionally, this metric can be used in many fields beside image processing where distance calculation is needed. Its massively parallel computing requirement can also effectively be implemented on the CNN architecture.

The generalized methodology for 2D object shape comparison demonstrates that the use of spatio-temporal dynamical processes extend the possibilities of this research area and several ways are opened for new type of metrics.

4 USE OF SPATIO-TEMPORAL DYNAMICS IN ALGORITHM DESIGN FOR MODEL BASED OBJECT CLASSIFICATION

In this chapter it is discussed how spatio-temporal dynamical processes can effectively be used in object classification algorithms. In the center of interest the special so called non-equilibrium algorithms on data flow are placed using different constrained wave propagation based operations. This part of my work can be considered as the extension of the use of technique that was used in wave based metric computation (*Chapter 3*). The focus is put on how constrained wave propagation and its time development coded into spatial maps can serve as elementary operations for object's feature extraction, model generation and classification based on wave metric computation.

As a possible application of spatio-temporal dynamical processes the experimental results of bubble-debris classification in condition based maintenance will be presented. It is shown that the algorithm can be realized on already available CNN chips. Furthermore, it is argued that for a real-time solution, templates and methods of the complete algorithm belong to the implementation frame that is considered for the next generation of CNN Universal Chips. This study demonstrates that the discussed novel approach allows a reliable classification of debris particles in real-time using a spatio-temporal CNN visual microprocessor.

4.1 Spatio-temporal Dynamics as Elementary Operators in Object Classification Algorithms

Probably the most challenging future direction in signal processing will be to provide efficient computational and algorithmic support to the emerging sensor technology. Recent advance in fabrication processes and material technologies made it possible to design different sensor arrays detecting tactile, temperature, chemical, and other physical and biological signals. Like in image processing these sensors produce continuously thousands of signals and these signals should be processed. Recent development in sensory devices can be viewed as the third revolution process in the electronic and computer industry. The statement is that a new computing paradigm is needed [7]. Processing analog array signals, an efficient analog sensor computer (or spatio-temporal computer) is necessary and the CNN-UM architecture seems to be an ideal candidate to play this role. Currently, extensive researches are going on to combine sensory technology with computing devices. Many experts do agree that the framework of stored program analogic signal processing or analog cellular computing offers one of the most promising computing technology for dealing with a huge number of input signal channels in real-time. The targeted goal is to produce tools with on-chip merged of different types of sensory arrays and computing devices. Such devices will be in the near future see, touch, smell and sense the surrounding world and will be capable of making quick decisions depending on variant events. This would positively influence different field of technologies like in safety control (e.g. monitoring systems in automobiles, traffic control), quality assurance (control in manufacturing, in food industry, in medicine, etc.), and other fields of industry.

Into the CNN-UM architecture different sensory interfaces can be embedded and 2D signals can efficiently be analyzed by partial differential equation (PDE) based methods [96], [97], [98]. PDEs describe diffusion and wave-type processes that can be constructively used in computation. The CNN-UM architecture is an ideal hardware to approximate these process, thus provides a framework for building 2D-flow processing machines. There are already many results on this subject [21], [23], [24], [25], [26]. Since the developed algorithms have been optimized for vision flows novel computing techniques are also needed to extract, reconstruct and manipulate different, specific types of information. The CNN-UM architecture viewed as a PDE-machine makes possible to develop these novel nonlinear wave-type computing techniques where they use both equilibrium and non-equilibrium operations. It is an exciting challenge how to combine these diverse phenomena of sensory arrays in useful algorithms running on a standard spatio-temporal computers.

4.1.1 Wave Type Processing on Spatio-temporal Computers

The signals to be processed are two-dimensional (2D) signal array or image flows. The 2D array may represent pictures or any other 2D sensory output signals of continuos value in continuos time. Without loss of generality they can be treated as image flows with the only constraint that they are discretized in space. The formal framework of analogic computing on image flow is as follows [8], [9].

An image flow is described by a function $\Phi(t)$: $\{\Phi_{ij}(t)\}$ of two discrete spatial coordinates and the continuos time coordinate. An image flow or a video flow is defined as the evolution of an image $\Phi(t)$ in a given interval $t \in T=[0, t_d]$. Generally an image at a given time contains either gray scale values or appropriate color components (e.g. R, G, B values). A spatio-temporal

elementary instruction on this image flow can be defined as $\Phi_{out}(t):=\Psi(\Phi_{in}(t))$, where $\Psi(.)$ is a function or functional on image flows. This function produces usually settled output so it operates as a mapping from the input image to the output equilibrium image. Loosely speaking, the usual analogic algorithms belong this class, i.e. they use *equilibrium type spatio-temporal instructions*. This is the case if partial differential equation (PDEs) based algorithms are implemented and settled outputs are considered as results. The another way of investigation is the study of flow processing algorithms that exploit transient behavior of dynamic processes. The idea is the following. Starting from the original image flow and applying the spatio-temporal dynamical process acting and evolving on this flow we stop the transient without waiting for the steady state. This new image flow will be the input data for the next dynamical operator and sequence of different dynamic operations would produce the *non-equilibrium type spatio-temporal algorithm*. Such an algorithm might consist the following type of operations:

- equilibrium type spatio-temporal operations (e.g. settled output of a PDE),
- non-equilibrium type spatio-temporal operations (e.g. stopped transient of a PDE based operator),
- spatial logic instruction on pictures (snapshots of an image flow) acting on pixel-wise,
- spatio-temporal combination of image flows pixel wise.

While algorithms of digital computers are defined mathematically via the μ-recursive functions, the analogic spatio-temporal algorithms are defined via the α-recursive functions. Computational complexity has been thoroughly studied for μ-recursive functions on *integers* and they are directly related to standard digital computers. Computational study on *reals* [39] has shown their limits if numerical algorithms on *reals* are considered. Spatio-temporal algorithms, however, are continuous time and continuous value operations on flows. The start-up study shows its analysis and the relation between previous approaches [8].

4.1.2 As Simple as Possible

One might think that the use of nonlinear dynamical processes in non-equilibrium spatio-temporal algorithms is too sophisticated and it is only to complicate matters. Why do not we use already well-known and tested methods and try to adapt them for the problems? However, it is argued that data processing on image flow based on nonlinear dynamical processes is quite natural. There is not signal conversion between different representations, processing takes place near the sensors, and operations are also natural. Defining "natural operators" would go beyond the scope of this work but heuristically it is quite understandable. The world around us behaves in this way and we describe its dynamical properties in mathematical equations. This dynamic treatment is somehow approached by this methodology. If we describe such an algorithm we can use terms like stretching, smoothing, amplifying, extracting, propagating, etc. All these functions can be exactly defined as PDEs and wave type operations. The term "*as simple as possible*" refer this advantage. We can combine these "simple" operators into very sophisticated algorithms and solve difficult problems. Up to know to program these "simple" operators was the most difficult task using digital computers but for an analogic sensor computer these operators are its elementary operations, i.e. to solve a partial differential equation or produce different wave type phenomena. The CNN-UM architecture provides a real framework to investigate this possibility. The next section will address this subject for object classification algorithms i.e. how spatio-temporal dynamical processes can efficiently be used in classification problems. Attempt will be made to construct such classification algorithms where these elementary nonlinear dynamical processes play the major roles.

4.1.3 Object Classification Using Spatio-temporal Dynamics

There are several problems concerning a large number of objects that should be continuously classified or compared to their references in real time. It is important to solve this type of problems using only a few numbers of operations which should be as simple as possible. In this case it is usually impossible to extract many features and combine them into a vector as well as to classify objects computing differences between these vectors. Time constraints enable often only *one measure* and *one decision*.

In *Chapter 3* we have seen how a spatio-temporal dynamical process (trigger wave type propagation) can effectively used for object's comparison. The basic idea was to record time evolution or development of this wave propagation and information converted into a spatial map can be used for different investigations. This concept can be extended to steps of classification processes as well. Not only difference measurement among objects but also the necessary model or reference generation can be done via constrained wave type propagation. Based on these considerations, an algorithm of this kind generally can be partitioned into three major parts as follows.

- Compute some basic features of the objects. For instance, central points and sizes of the objects might serve as primary data.
- Generate models using both the previous and *a priori* information. An example for a priori information is the expected structural properties of the models.
- Finally, the comparison and classification using the wave type metric.

Emphasizing again that the computation should be fast, the same types of operators are used, namely, the computation on the recorded time evolution of propagation of trigger waves. The identical types of operators make it possible that a given physical implementation might have low complexity.

Without loss of generality examples on spatio-temporal dynamical processes will use the concept of the CNN-UM architecture. The spatio-temporal operators correspond to elementary operations of the analogic CNN computer architecture therefore fast computation is easily achieved. Based on this framework a two-layer structure is defined. The layers are identical but have different assignments. The first layer serves to generate wave type processes and these processes are used to extract features, generate models and explore objects for metric computation. The task of the second layer is to provide a spatio-temporal control for these nonlinear dynamical processes or record their time evolution for a further investigation. This general structural scheme shown in Figure 4.1 exploits the main advantage of spatio-temporal dynamical process based computation that it can be very fast. The computation comes out through the transients of processes. The combination of such operations embodies the spatio-temporal algorithm.

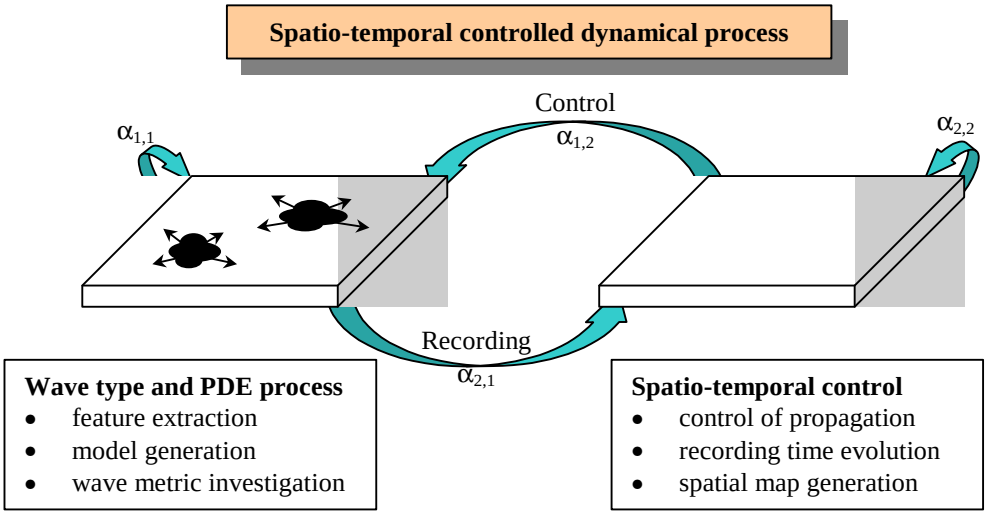


Figure 4.1 General architecture for spatio-temporal controlled nonlinear dynamical processes. Computation is based on wave type propagation and PDE mechanisms.

A spatio-temporal algorithm can be described on this structure now if we specify the input flow and operations ($\underline{\alpha}$) among layers.

$$F_{out}(t) = \ddot{\cdot}\ (F_{in}(t), \underline{a})$$

These operations using CNN-UM architecture is coded into template interactions. As it was mentioned before the first operational 2^{nd} order or three layer CNN architecture will be fabricated in the near future [27] and this architecture will be adequate such type of processes.

Next the following terms will be used. The pixel positions of the two layers will be referred as cells. These cells can contain different amount of charges that code intensity values. The α operators encode different wave type or PDE processes in their convolution kernels. The interaction between layers can be either linear or nonlinear connections.

4.1.3.1 Feature Extraction of Objects

As it was mentioned before a classification algorithm based on model comparison requires some information about unknown objects. Therefore features of objects might contain only main properties of objects respecting that analysis should be restricted to major characteristics. Location of objects and their sizes will be determined. It should be emphasized that following methods works on images or image flows and these inputs might contain several objects. The processing will take place on these objects parallel and simultaneously i.e. many objects can be processed at the same time.

Before examining the feature extraction possibilities the Distance Map generation via Distance Transform should be briefly analyzed.

4.1.3.1.1 Distance Transform

The distance transform is an operator applied to either gray scale or binary image and produces gray scale image [37]. The gray scale intensity values in the image encode the distance

of the pixel to the closest boundary point. There are several methods to compute a distance map of an image [82]. One way is to simulate a slow fire starting at all points on the boundary of a foreground region and propagate into the interior [83]. A value is assigned to each pixel, namely, the amount of time that the fire took to first reach that point. Another way to compute a distance transform is to perform multiple erosions with a suitable structuring element until all foreground regions of the image have been eroded away. Each pixel is labeled with the number of erosions that had to be performed before it disappeared. There are several different sorts of distance transform depending upon which distance metric is being used to determine the distance between pixels. The three most important cases will be considered. The Euclidean Distance is the length of the straight line between two pixels, given by using Cartesian coordinates:

$$d_E(A,B) = \sqrt{\sum_{i=1}^{2}(A_i - B_i)^2}$$

City Block Distance also known as the Manhattan distance. This metric assumes that in going from one pixel to the other it is only possible to travel directly along pixel grid lines. Diagonal moves are not allowed. Therefore the city block distance is given by:

$$d_{MC}(A,B) = \sum_{i=1}^{2}|A_i - B_i|$$

In Chessboard Distance the diagonal move is also enabled and counts the same as a horizontal move. This means that the metric is given by:

$$d_{Ch}(A,B) = \max_{i=1}^{2}|A_i - B_i|$$

The last two metrics are usually much faster to compute than the Euclidean metric and so are sometimes used where speed is critical but accuracy is not too important. In our task the first and second distance measures play role the third would result in false distances for model generation.

The Distance Transform is a very useful tool for different applications. It can be used for skeletonization, binary morphology operations (erosion and dilation), medial axis transform, fractal dimension measurement, and watershed segmentation. For binary morphology operation, *Distance Map* (DM) might be used as follows. Simple thresholding of the distance map can be used to select pixels that are farther from the edge than any desired extent of erosion. Similarly, the distance map of the background can be thresholded to perform dilation. Both of these operations can be carried out without any iteration. The distance map might be constructed non-iteratively as well, so the execution time of the method does not increase with object size (as do classical erosion methods) and is preferred to for large objects. Classic erosion applied to a circle will not shrink the circle uniformly but depending on the structuring element it will proceed at a faster rate in the diagonal or vertical direction. To define a really isotropic neighbor pattern the elements should be between 0 and 1 approximating $\sqrt{2}$, the distance ratio in Euclidean sense. Still, the repetitions of erosion operation cannot be avoided. These two problems might be overcome using Distance Map. An example is presented in Figure 4.2 containing two circles connecting together.

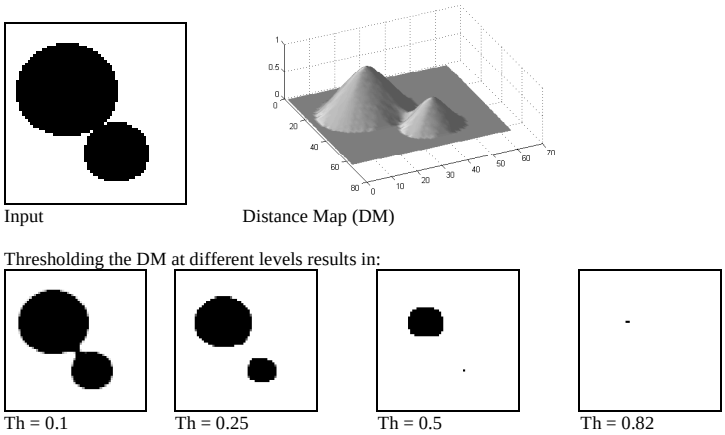


Figure 4.2 Erosion operation based on Distance Transform.

The advantages using Distance Map for morphology operations are

- Two operations are needed only (generation and threshold), instead of several morphological operations,
- Isotropy can be achieved.

The implementation of the Distance Transform on the architecture shown in Figure 4.1 is straightforwardly. On the first layer objects will be the initials of a wave propagation which shrinks objects continuously until they disappear. This is coded into $\alpha_{1,1}$ as a convolution kernel. Below three different possibilities are shown.

0.41	0.59	0.41
0.59	2	0.59
0.41	0.59	0.41

$\alpha_{1,1}$: Euclidean

0	1	0
1	2	1
0	1	0

City Block

0.5	0.5	0.5
0.5	2	0.5
0.5	0.5	0.5

Chessboard

In a CNN implementation these kernels stand for feedback operators (A template) and an additional bias value is also necessary. This bias value determines that the propagation is either global or local and its direction is expanding or shrinking. Thorough discussion about bias setting and its control role in propagation can be found in [26].

While trigger waves shrink down objects on the first layer the second layer records this propagation through $\alpha_{2,1}$. This is a specified value that is used to fill cells of the second layer in positions where objects still have points on the first layer. At the end of the process cells of the second layer will contain different values proportional to the time that the wave front took to reach that position.

4.1.3.1.2 Local Center Points of Objects

For feature extraction Distance Transform seems to be an ideal tool. The locations of local maxima and ridges (singularities of curvature) encode information about object location and size. We define the term *local center points* of objects as follows. If an object or object group (connected objects) can be constructed via combination of geometrical primitives then center points of such primitives consists of local center points of the object or object group. Geometrical primitives are defined via Distance Transform of objects. If the Distance Map of an

object contains a single global maximum then its position can be treated as the center point of the object. Such an object might play the role of the geometrical primitive. Examples for such geometrical primitives are shown in Figure 4.3.

Figure 4.3 Center point detection via Distance Transform of geometrical primitives.

If an object consists of more geometrical primitives then its Distance Map will contain more local center points. This is shown in Figure 4.4.

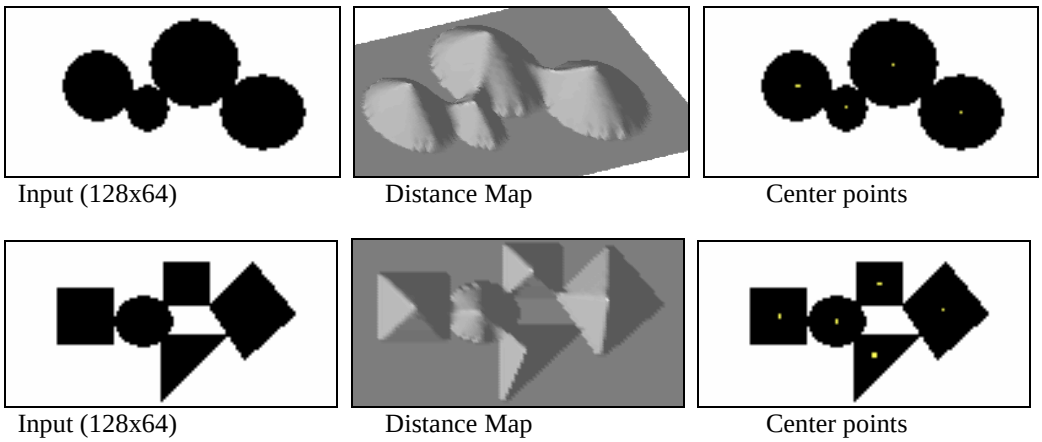


Figure 4.4 Center points of objects consisting of more geometrical primitives.

4.1.3.1.3 Medial Axis Transform and Skeleton of Objects

If Distance Map of an object instead of containing single maximum contains ridges then we can construct Medial Axis Transform (MAT) of the object via its Distance Transform. The Median Axis Transform is a gray-level image where each point on the skeleton of the object has an intensity which represents its distance to a boundary in the original object. The skeleton is also constructed from the Distance Map. The skeleton lies along the singularities (i.e. creases or curvature discontinuities) in the Distance Transform. This is used to calculate the Median Axis transform since the MAT is the same as the distance transform but with all points off skeleton suppressed to zero. Figure 4.5 shows an example.

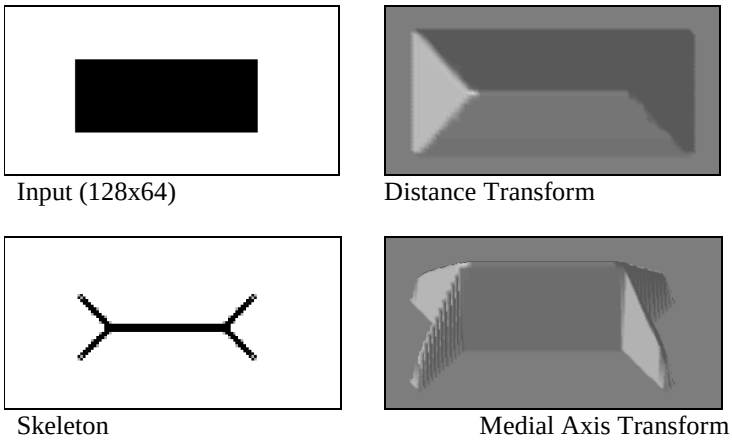


Figure 4.5 Medial Axis Transform and skeleton of an object via Distance Transform.

Just as there are many different types of distance transforms there are many types of skeletonization algorithms and Medial Axis Transforms all of which produce slightly different results. However, the general effects are similar. The skeleton provides a simple and compact representation of shape that preserves many of the topological and size characteristics of the original shape. In addition, to this, the Medial Axis Transform (not the pure skeleton) has the property that it can be used to exactly reconstruct the original shape if necessary.

4.1.3.1.4 Radii Map of Objects (Size Estimation)

The second important feature is the size of the objects. There are more candidates for this job. The Medial Axis Transform can serve this task because its intensity values encode distances along object's skeleton. Another possibility is to use as Radii Map of objects the intensity values at local center points in the Distance Transform. Figure 4.6 shows these possibilities for Radii Maps of objects.

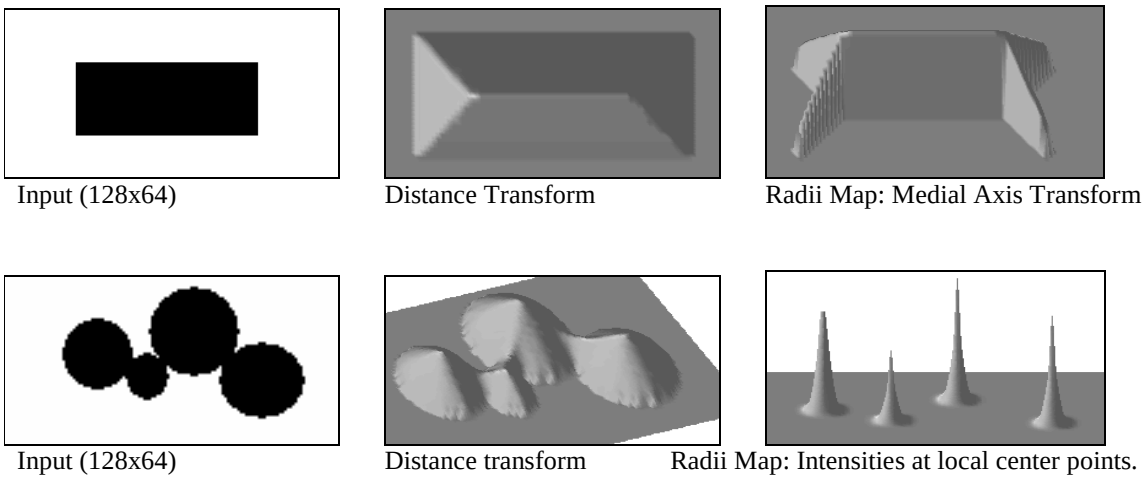


Figure 4.6 Radii Map of objects constructing from the Distance map.

4.1.3.2 Spatio-temporal Controlled Model Generation

Center Points and Radii Map are the extracted features and this information is used to generate models for each object. This process will also work parallel on objects. The general scheme of model generation is that Center Points of objects are initials for trigger waves to expand patches on the first layer while the Radii Map of objects controls this propagation on the second layer via $\alpha_{1,2}$. The wave propagation coded in the $\alpha_{1,1}$ determines the shapes of models. It can be isotropic or anisotropic propagation as well. Implementation details on a concrete example will be presented in the next section.

4.1.3.3 Classification Using Nonlinear Wave Metric

Finally the third stage of the classification process applies nonlinear wave metric to classify objects. Its method was thoroughly discussed in *Chapter 3* and the next experimental study will demonstrate its advantages.

4.2 Condition Based Maintenance of Engines

In this section, I present initial results of cellular neural network based nonlinear Hausdorff metric to high speed pattern recognition of gray scale images. The application is to a problem involving separation of metallic wear debris particles from air bubbles. This problem arises in an optical-based system for determination of mechanical wear. This research focuses on distinguishing debris particles suspended in the oil flow from air bubbles and aims to employ CNN technology to create an on-line fault monitoring system. For the class of engines of interest bubbles occur much more often than debris particles and the goal is to develop a classification system with an extremely low false alarm rate for miss-classified bubbles.

The designed analogic CNN algorithm detects and classifies single bubbles and bubble groups using binary morphology and nonlinear Hausdorff metric. The debris particles are separated based on nonlinear Hausdorff distances computed between bubble models and the unknown objects. Initial experiments indicate that the proposed algorithm is robust and noise tolerant and when implemented on a CNN Universal Chip it provides a solution in real time.

Two types of implementation will be presented at each sub-algorithm. One of them is an iterative type (based on repetitive application of morphological operations) while the other one is called dynamic type method (transient of a wave propagation result in the operation). The importance of the first method is that it can already be tested on present VLSI implementations of CNN-UM architecture. This phase of work is important to test parts of algorithm on existing CNN chips and determine robustness of the proposed algorithms. The dynamic type implementation is much faster but supporting CNN chip architectures are still under design nevertheless they will be applicable in the near future [27].

4.2.1 Preliminaries and System Requirements

In many sophisticated systems, it is important to obtain information about the state of a subsystem via mechanical wear analysis. System malfunctions need to be predicted and an appropriate alarm signal should be generated for the operators. This is the case in our specific problem where the system to be investigated is an engine. To perform the inspection optical sensing and subsequent image analysis was chosen because the strong mechanical vibration and electromagnetic interference render this problem very difficult to be solved using other types of

methods. The main task that arises in the optical detection of mechanical wear debris in highly aerated lubrication flows is to distinguish air bubbles from debris particles. In order to detect sudden changes, running an on-line monitoring system instead of an off-line one or monitoring from time to time is inevitable.

The basic idea behind the analysis is the comparison of all particles to bubble models (circles and overlapping circles) and the classification based on a difference error measurement. The problem of distinguishing debris particles from air bubbles is difficult due to the coarse resolution of the images and to the requirement for an extremely low false alarm rate for miss-classified bubbles at a very high processing speed. It will be shown that CNN technology is a promising candidate to solve this problem which requires high-speed image processing and compact design.

System Specifications and the Present Test System

At the NRL (Naval Research Laboratory) a bubble-debris classifier is being developed. Figure 4.7 shows the test environment in which the imaging system is applied. The imager device is located in the lubrication flow system which has a flow speed up to 10 m/s. A pulsed laser illuminator projects an image of the fluid along with various suspended objects onto a CCD sensor. These images are to be processed and analyzed by the system. The present test system processes *binary images* of 512x512 pixels on line at frame rate 500 frames/sec, using a high speed serial processor. More details of the present system are provided in [99], [100].

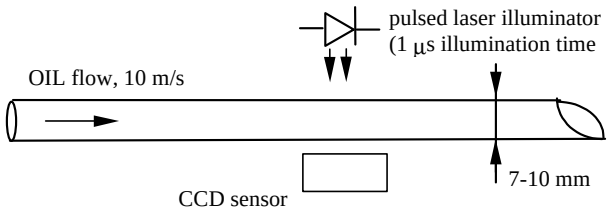


Figure 4.7 Optical sensing setup for the condition monitoring system. Frame rate can be as high as 1000 frames/sec.

Figure 4.2 shows a typical image to be processed. This image contains debris particles and air bubbles some of which overlap due to occlusion. The major goal is to distinguish debris particles from the air bubbles.

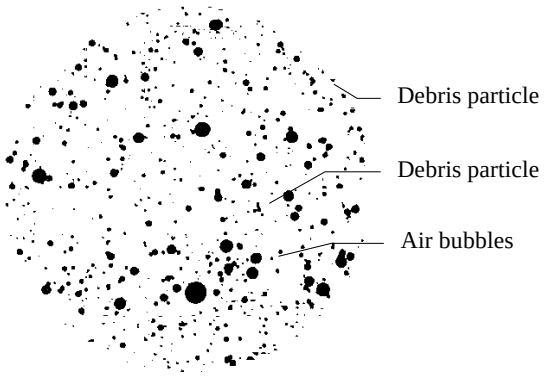


Figure 4.8 Typical image of the oil flow with floating particles as recorded by the CCD sensor. The image size is 512x512 and the pixel size is about 14μm.

Figure 4.9 shows the density and size distribution of various objects present in the oil flow. For the generation of alarm signals only objects larger than about 50 μm are interesting because these include cuttings and worn off particles that provide information about mechanical wear-off. Because the size-ranges of different objects overlap it is not enough to only determine the size of an object, but classification should be carried out based also on its shape, and features such as aspect ratio, external compactness and bending energy.

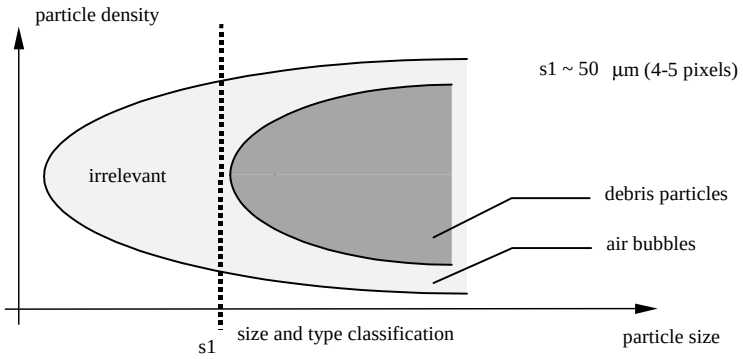


Figure 4.9 Particle size/density distribution in the lubricant. Small debris particles are excluded because they relate to a benign wear condition.

The approach followed at NRL was to tune the system parameters to achieve the desired false alarm rate and to minimize the percentage of miss-classified debris particles. The system operates as a series of rejection filters with increasing time consuming classification tasks applied to a smaller and smaller number of objects. The strategy of using a sequence of tests for increasing computational complexity was determined by the limitations of the standard (digital) single CPU computer. This approach makes the classification algorithm difficult to tune in that the thresholds used for one test can impact on subsequent tests.

Proposed CNN-based Solution

Figure 4.10 shows the setup of the proposed architecture. A significant advantage offered by the CNN is that the image processing takes place close to the optical sensor alleviating the need for high throughput data transfer and only low-bandwidth, high relevance data is transferred

from the sensor that is easily tackled by conventional digital hardware. This results in significant speedup in processing and decreases the size of the apparatus that mounts directly on the gearbox.

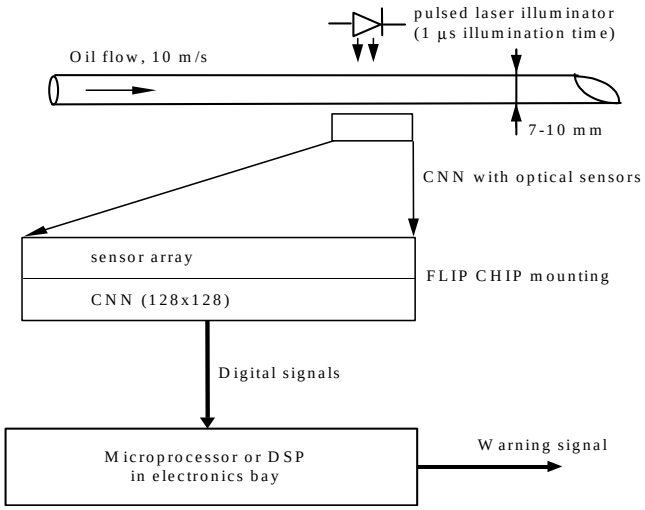


Figure 4.10 Outline of the proposed CNN-based solution. By employing image sensors mounted directly on the CNN chip, sensing and high-volume processing can be performed at the same place.

The CNN processor can be used to solve the following tasks:

A) size classification (to filter too small objects)
B) air bubble / non-bubble detection (to filter extraneous objects)
C) fault type classification (determine type of wear particles)

Figure 4.11 shows the block diagram of the proposed approach.

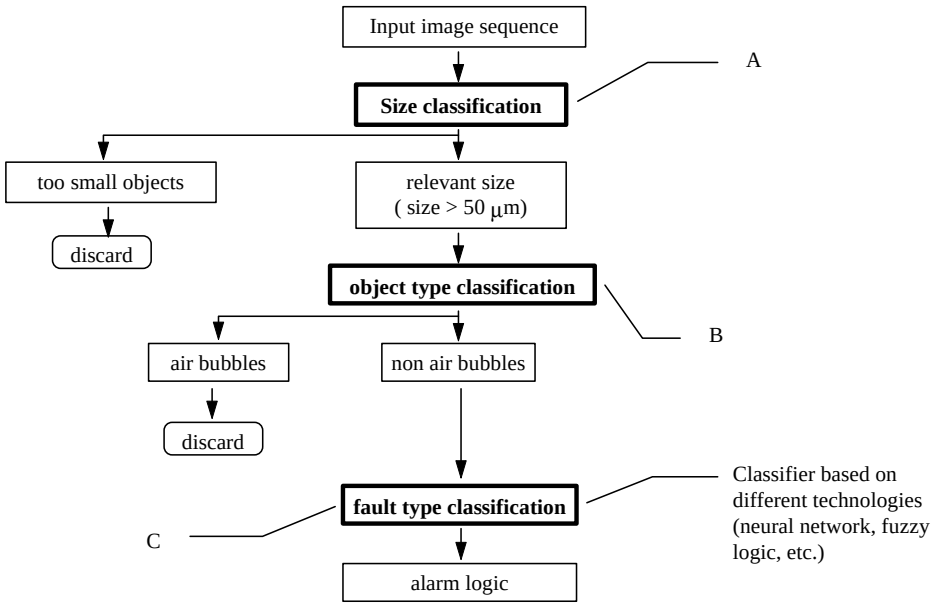


Figure 4.11 Block diagram of the processing of oil flow images

Elements of CNN Solution - Size Classification

There are well-studied methods for size classification in the CNN [28-30]. This is part A of the processing pass. These methods usually employ a series of peeling and comparison operators as well as local logic operations to identify binary objects of specific sizes. These are directly applicable to this problem and can be used with little modification. Fault type classification (part C) is not addressed in this work because it is not essential for generation of an alarm signal in an online monitoring system. This would be necessary for a subsequent analysis of debris particles. We will focus directly on object type classification (part B).

Roughness measurement

Next, some possible methods are presented to measure the surface roughness of objects. This computed quantity can be used at the second stage of processing (object type classification) and at the third stage (fault type classification), as well. Figure 4.12 shows the results of three different methods for computing surface roughness.

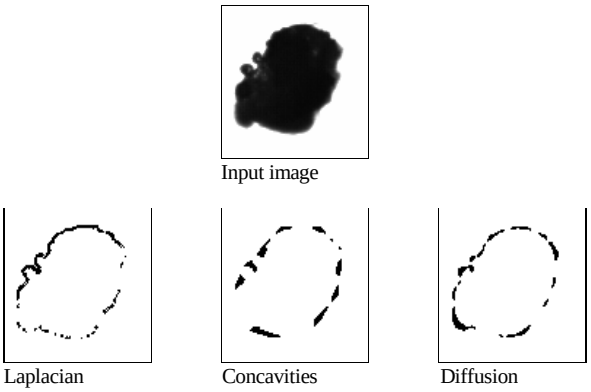


Figure 4.12 Techniques for determining surface roughness: the methods used a Laplacian operator, concavity detector, and diffusion operators, respectively. This information serves as input data for wametric based roughness measurement.

This techniques might produce input data for wave based roughness measurement discussed in Section 3.5.

Using Laplacian

First the Laplacian of an image is computed and the pixels in this image are squared (this can be approximated by the absolute value function that gives a close estimate in the interval considered). This image is then thresholded, and converted into a binary image. For instance, the number of connected components of this image might be a measure of roughness of the boundary of the original object. Figure 4.13 shows the steps of the algorithm.

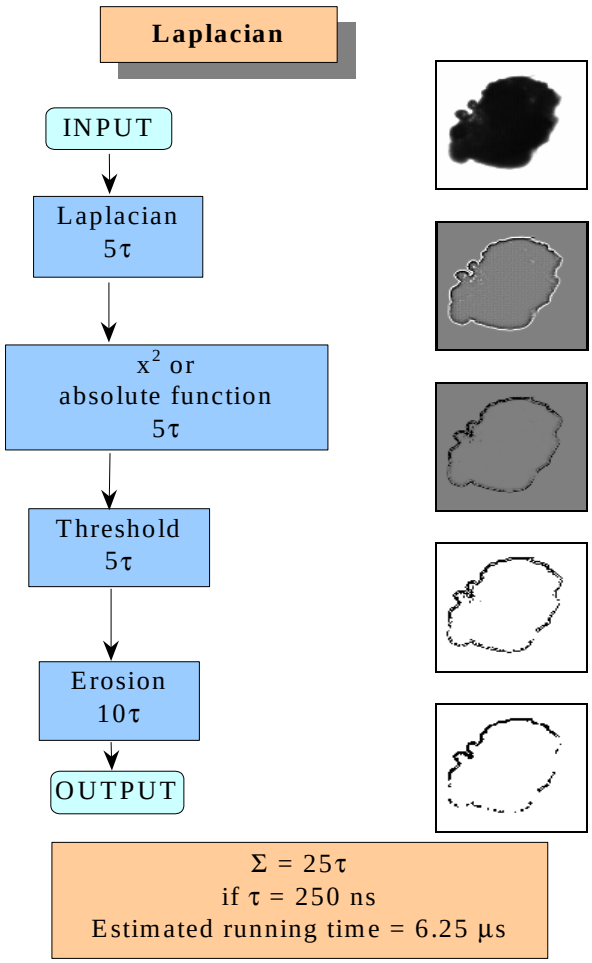


Figure 4.13 Measurement of surface roughness based on the Laplacian operation.

Finding Concavities

The basic idea here is to find the concave parts of objects. First the gray-scale image is converted into binary image via thresholding operation. Next, pixels are driven to black which are located at concave places using the "hollow" template. This template turns black all those white pixels which have at least four black direct neighbors. We call concave those white pixels which are surrounded by black pixels from at least four of the eight possible directions. The network transient must be stopped after a given amount of time depending on the size of the largest holes to be filled in. Next we extract concavities of objects using logical XOR operation between the thresholded image and filled image. Figure 4.14 shows the steps of the algorithm.

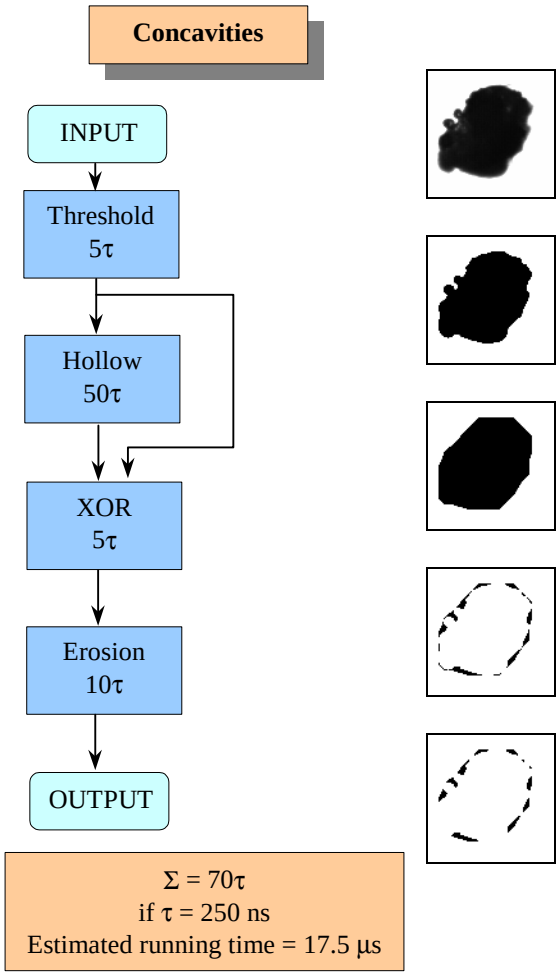


Figure 4.14 Measurement of surface roughness based on finding concavities.

Applying Diffusion

This method computes surface roughness using a diffusion operator. The image is smoothed in order to fill the rough parts of the object and between this and the original a difference is computed. First the image is thresholded, then diffusion is applied to the thresholded image. The diffused image is thresholded again and logical XOR operation is applied between the two thresholded images. Spurious isolated pixels are removed by erosion. Figure 4.15 shows the steps of this algorithm.

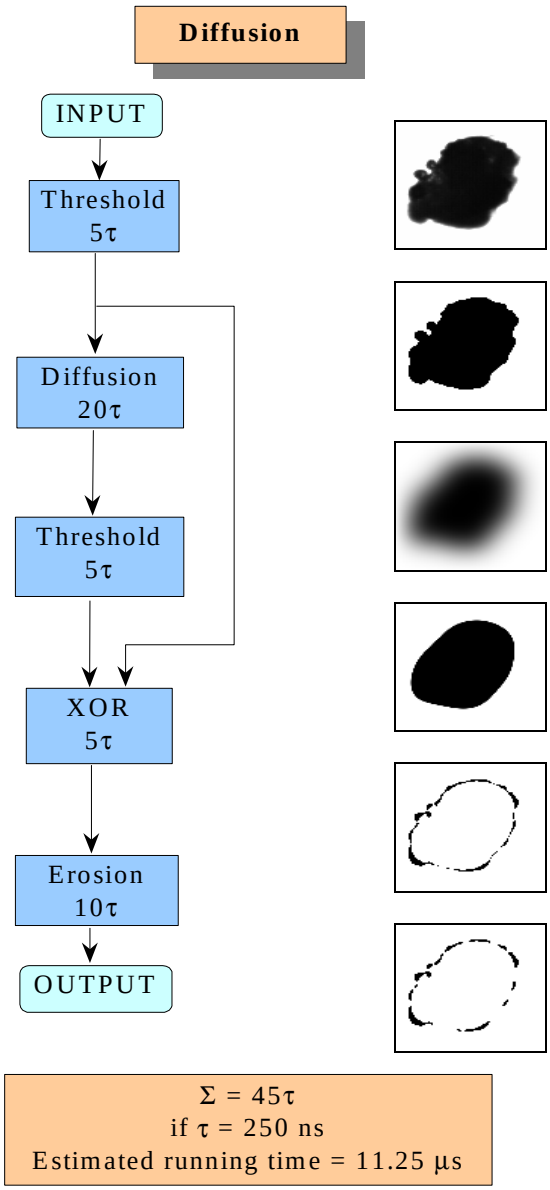


Figure 4.15 Measurement of surface roughness based on diffusion operation.

Comparison of Methods Measuring Roughness

The first method using the Laplacian of the image is the fastest of the three approaches studied here. Its implementation, however, requires the use of a local arithmetic unit for pixel-wise multiplication of the image. This feature, while feasible, may not be available in the first CNN processors. Also, the quality of the output seems to depend on the local contrast of the edge contour. The other two methods involve only simple 3x3 linear templates and local logic (XOR), the functionality readily available in CNN arrays. They both seem to produce robust output and are easily adaptable to varying scales (object sizes). This is desirable since the resolution of the input image may vary with the particular imager used [30]. For a qualitative comparison of these different methods measuring surface roughness an artificial object sequence is need to be created with different shapes and curvature roughness. This is still an ongoing research.

The presented methods can be considered as the first step toward measuring the surface roughness of the objects. In a more elaborate algorithm the detected pixels and the number of

connected components could be normalized with reference to the object's area. This way we would obtain two feature numbers. The normalized number of detected pixels referring to the variance of the shape and the normalized number of connected components corresponding to the variance of object's curvature. Based on these numbers a robust classification scheme could be developed (see Figure 4.16).

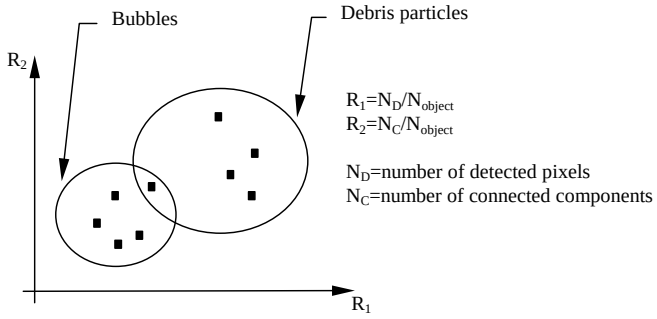


Figure 4.16 Illustration of the hypothesis that combining features from roughness measurements could be effectively used for classification. R1 and R2 are the computed features from the surface roughness measurements. The computed characteristics should be larger in case of debris particles than for the bubbles. The size of the intersections of the overlapping sets might be decreased using further methods. This could be exploited in a fuzzy algorithm.

Another possibility to measure surface roughness is the use of fractal dimensions. With proper simplification this can be implemented on CNN.

These characteristics and the results of wave metric based surface roughness measurement could be exploited in a fuzzy algorithm. Such a method seems to be necessary to arrive to a robust and reliable solution of this classification problem. It is important note again that this analysis is not necessary for object type classification. Fault type classification would require this more sophisticated investigation.

4.2.2 Bubble-debris Classification Using Morphology and Nonlinear Hausdorff Metric

The approach followed here is the filtering out air bubbles, using binary morphology [28], [29] and nonlinear Hausdorff metric[86], [88]. In the experiments we used binary images, the output of the field-programmed gate array that in the current system thresholds all gray-scale input images at a fixed level right after the acquisition. In a complementary study [112] we have investigated how the quality of these binary images can be improved by using a locally adaptive front-end filtering and segmentation strategy applied to the original gray-scale images.

Classification is based on object-model comparison via difference measurement. A model is a set union of circles based on the hypothesis that an object is a group of one or more overlapping air bubbles. Figure 4.17 shows the flowchart of the bubble-debris classification algorithm containing Feature Extraction, Bubble Model generation, and nonlinear Hasudorff Distance computation. The Feature Extraction block detects the center points of all objects and measures their sizes (*Radii Map*). These two characteristics are used for bubble model generation via propagating trigger waves. Both features are extracted from a gray-scale image called *Distance Map*. This map is generated from the input image by using an operator similar that generates the *Euclidean Distance Map*. The brightness of each point of an object in the *Distance Map* encodes the distance to the nearest point in the background. The distance slightly depends

on the measuring convention. The positions of local maximum values encode the center points of the objects (*Center Points*), while brightness values at local maxima encode the size of the objects to be modeled (*Radii Map*). The critical issue is to find the exact mass center of an object. In the case of multiple bubble groups, we need to obtain more than one center point. In the next step, trigger waves propagate from the center points and expand circles around them until the *Radii Map* stops this propagation. The nonlinear Hausdorff distance is computed between the bubble models and the input image in the third step. If the measured distance is large the object will be classified as a debris particle.

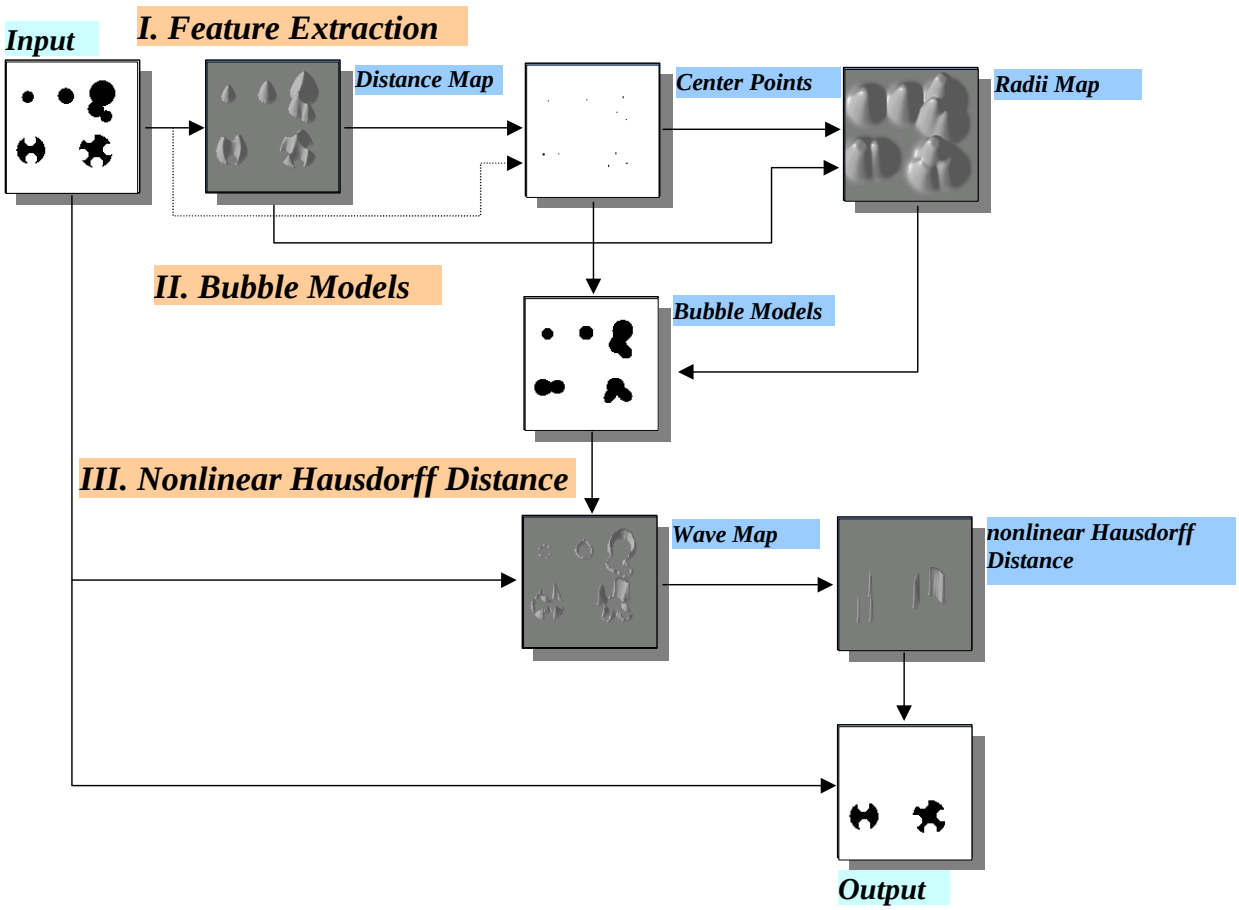


Figure 4.17 Flowchart of the bubble-debris classification algorithm containing Feature Extraction, Bubble Models generation, and nonlinear Hausdorff Distance computation. The artificial example contains objects in two rows. The first row contains bubbles or bubble groups and these objects should be classified as air bubbles. The second row contains wear particles to be classified as debris objects.

Comparison of this algorithm to the one used at the NRL shows a striking difference. This algorithm is classifies objects simultaneously in an image, in contrast to the other one in which each object is examined separately.

4.2.2.1 Feature Extraction

Two types of algorithmic implementation can be considered. One of them needs only binary morphology operations and the other one operates using wave propagation approach. The first method is an iterative process, therefore it is slower but existing CNN chips support these operations. The second one results in the fastest solution but needs a sophisticated hardware. In a CNN type implementation this requires a two-layer CNN structure.

4.2.2.1.1 Iterative type feature extraction

Next, the iterative type feature extraction will be discussed. The central part of the iterative method is the center point extraction and from this result the Radii Map is constructed through the Distance Map. The method to find the center points of the objects is as follows. A morphology operation called masked erosion is now defined. The image contains objects of different sizes that can be specified in number of morphological steps. Using a simple erosion procedure smaller objects would disappear before finding their center points while larger objects would still contain many pixels. A masked erosion approach can prevent smaller objects to disappear i.e. those sets which would vanish with the next application of the erosion operator. One step of masked erosion which is applied several times is shown in Figure 4.18.

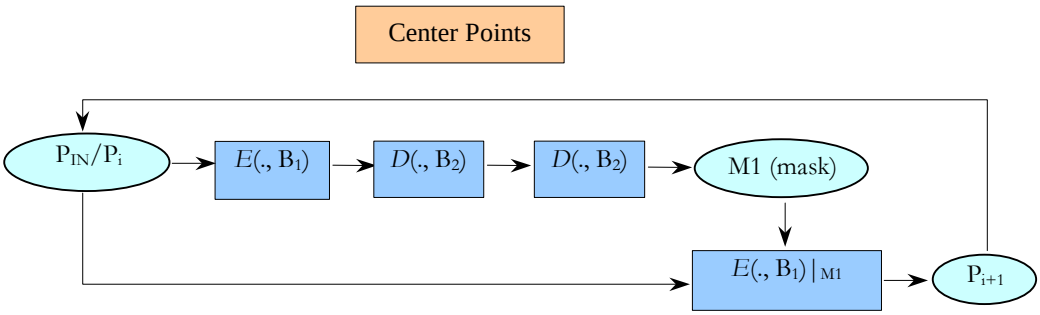


Figure 4.18 Iterative type center point detection of objects using binary morphology operations. B_1 and B_2 denote for structuring elements (kernels) used in erosion - *E(.)*, and dilation - *D(.)*.

The mask image (M1) is a black & white image where black pixels indicate those locations where a morphology operation might have an effect. The changes of pixels that are located at white mask pixel positions are prohibited. In our case, the mask should contain only the larger objects which won't totally disappear during an erosion i.e. the erosion operation may have effect there while at small objects should be prohibited. The mask is generated in three steps. First, the erosion operation deletes smaller objects. After that the two dilation operations expand the objects larger than their original size. This mask image (M1) enables the erosion only at locations of larger objects.

In a CNN implementation the template of erosion used here is (B term contains the kernel)

A=

0	0	0
0	2	0
0	0	0

B=

0.5	1	0.5
1	1	1
0.5	1	0.5

z = -6.5

The template of dilation is

A=

0	0	0
0	2	0
0	0	0

B=

1	1	1
1	1	1
1	1	1

z = 8.5

It is important to note that for dilation a slightly different kernel should be used than for the erosion. The dilation should be repeated at least twice. The reason for that lies in the fact that the erosion operation is not invertible in general (e.g. if we use the same kernel for dilation as for the erosion we should apply dilation at least three times otherwise false center points will appear). Based on a priori information, we know the approximate sizes of the largest objects and it is possible to estimate the maximal number of steps that should be used for the masked erosion in order to get a picture containing the centers of the objects.

While center points are computed the radii information of objects is also extracted. Figure 4.19 shows the block of Feature Extraction in details. The Radii Map resembles the Distance Map although they are slightly different.

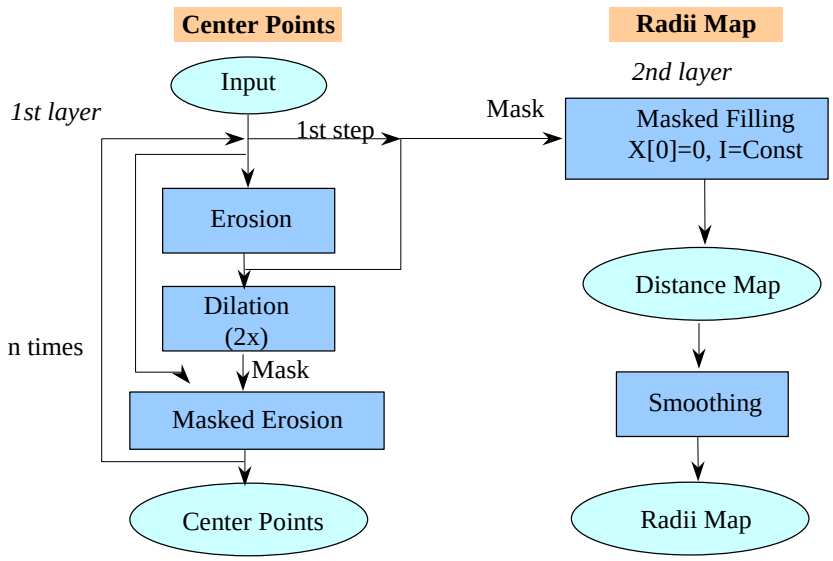


Figure 4.19 Iterative type implementation of the Feature Extraction block which detects the center points of the objects and measures their sizes constructing two images, namely the Center Points and the Radii Map.

The steps of constructing Radii Map are as follows. A layer will be filled with constant current and the voltage levels of cells will indicate the sizes of the corresponding objects. We use the results of process of center points detection. The output of the first erosion is used as a mask to control this filling. This mask will contain smaller and smaller patches around the center points as erosion goes on and shrinks the objects in several steps. At the end of the process the state levels of cells at locations of center points will have the largest values and gradually decreasing around them. The largest object results in the largest cell value at its center point. This map is smoothed resulting the Radii Map.

4.2.2.1.2 Dynamic type feature extraction

The steps of the dynamic approach in a two-layer CNN structure are as follows. First a gray-scale map, called *Distance Map*, is constructed in a two-layer process. The state of the 1-st layer contains the input image and at the end of the process the second layer will contain the distance map. The image on the first layer is eroded continuously while on the 2-nd layer cells are filled with constant current. The continuous erosion means that objects are shrunk with a propagating template and at the end of the transient all objects will disappear. The process of Distance Map generation is shown in Figure 4.20.

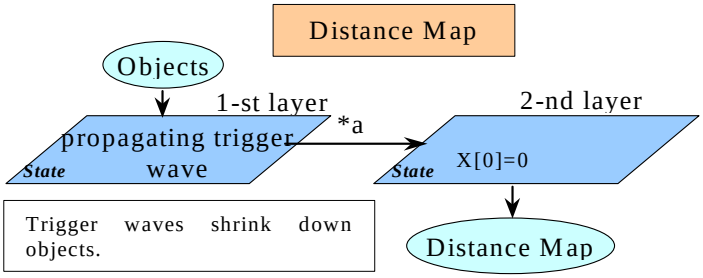


Figure 4.20 Distance Map generation using wave phenomena. Trigger waves shrink down objects while cells are filled in a second layer to record time evolution of the wave propagation. This second layer contains finally the Distance Map.

The template for erosion on the first layer is:

$A_{1,1}$=

0.41	0.59	0.41
0.59	2	0.59
0.41	0.59	0.41

z = -6.5

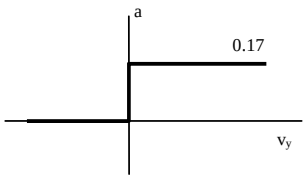
Template for the second layer is a simple threshold driving from the first layer.

$A_{2,2}$=

0	0	0
0	1	0
0	0	0

$A_{2,1}$=

0	0	0
0	a	0
0	0	0



These templates together generate Euclidean Distance Map (EDM). Type of the Distance Map is determined by the template operating on the first layer. Changing the values in template $A_{1,1}$ will produce different Distance Map.

The positions of local maximum of *Distance Map* encode the *Center Points* while cell values at these points are related to objects sizes (*Radii Map*).

4.2.2.1.2.1 Center Points detection

The idea to extract center points from a Distance Map (DM) is as follows. A DM contains several points of local maximum. These points of local maximum are mainly located at position furthest from object's boundary and can be considered as local center or weight points. The term 'local' is used because these points are not theoretical weight points of an object. In simple cases like a circle, rectangle, or an uncomplicated shape, it gives the theoretical weight point. In more

sophisticated case it determines more points depending on the object's shape. These points can be considered as local center points and might be more useful in practical application than theoretical weight points. To extract these local center points GlobMax template [32] is applied to smooth the surface of a DM. The operation has to be stopped at a given time otherwise each cell of the layer will only contain the highest value. In the smoothed distance map local maxima are unchanged but pixels around them have higher values. This image is subtracted from the original DM and threshold operation at zero level will extract points of local maximum. The running time of the transient of GlobMax operation affects strongly the result of detected points. Figure 4.21 shows an example that how the running time of the GlobMax template modifies the detected points.

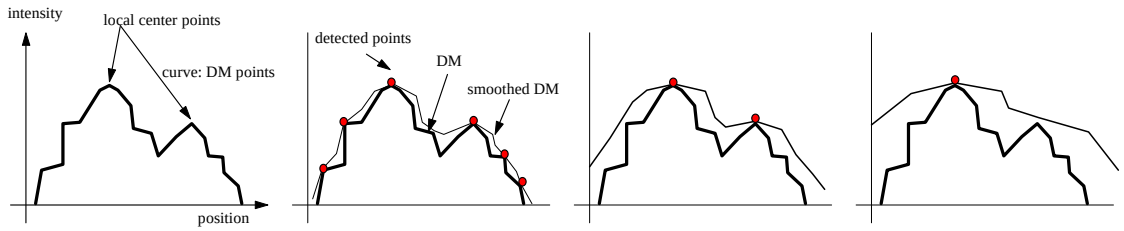


Figure 4.21 Detecting points of local maxima on Distance map using GlobMax operation.

The following pictures show examples using morphology operation or Distance Map based Center Points detection.

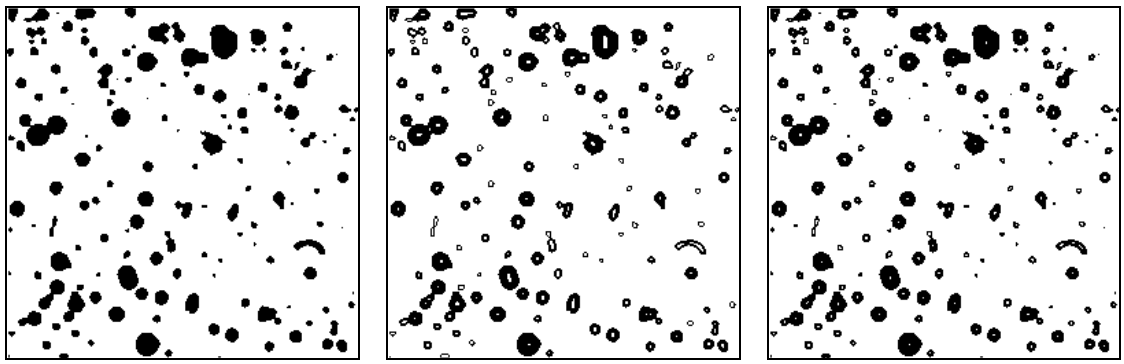


Figure 4.22 Center Points (CP) detection using iterative morphology or Distance Transform.

Comparison between iterative morphology and Distance Map based center point extraction is shown in Table 4.1.

Method of Center Point detection	Advantages	Disadvantages
Iterative Morphology	• dependence only on structuring element • robustness	• point group as center point • false center points • slow (iterative method)
Distance Map based center point extraction	• fast • single points represent center locations and no point groups	• possible undetected center points • parameter sensitivity • requires two layer implementation

Table 4.1 Comparison of different center point detection methods.

The possibility to improve the detection is the combination of different center point detection algorithms. The larger point groups can usually be filtered out. This means that at a given position the smaller point set is chosen as center point. Another possibility is to use different structuring element to produce distance map. Three kernels were tested.

0.41	0.59	0.41
0.59	2	0.59
0.41	0.59	0.41

D1: Euclidean

0	1	0
1	2	1
0	1	0

D2: City Block

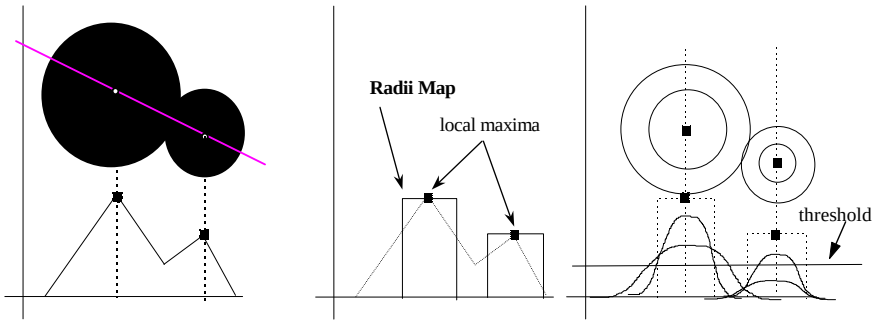
0.5	0.5	0.5
0.5	2	0.5
0.5	0.5	0.5

D3: Chessboard

The differences of generated center points were not significant therefore the Euclidean Distance Map was used for Center Points detection.

4.2.2.1.2.2 Radii Map generation

The Radii Map is designed to control model generation. The next sub-section will detail the bubble model generation, here, it is enough to know that a wave propagation will be controlled via Radii Map as a space-time control image. What candidates are for Radii Map? The main requirement is that for circle models an isotropic expansion is necessary. In the earlier experiments distance map was used as radii map and the generated bubble models were very similar to the original objects. However, the distance map is not proper candidate for radii map because it preserves the shapes of objects. A more suitable Radii Map can be constructed from Center Points and Distance Map. Around each center point columns will be grown with equal widths but different heights. The locations of center points determine the starting positions around GlobMax template will grow columns width heights determined by the distance map. The constructed radii map makes possible that all bubble models will be perfect circles. The next Figure shows the idea. Comparison of different Radii Maps will be presented in the next section with bubble model generation.



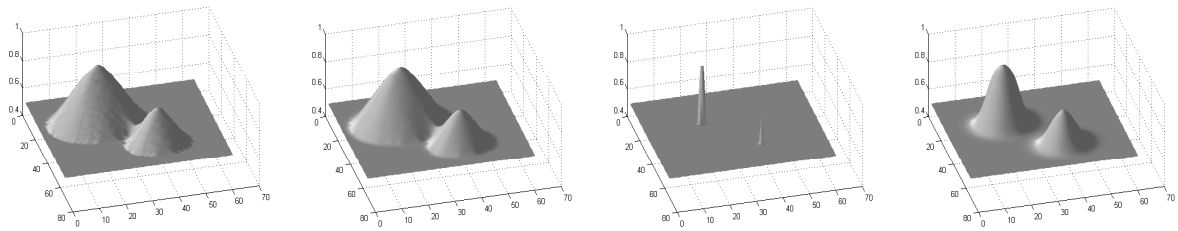
Two circles connected together and distance map intensities along the line. The intensities at local maxima (Center Points) are extracted.

Radial map as columns with equal widths but different heights. Globmax operation is applied at local maxima.

Smoothing radial map for controlling the shrinking of bubble models. Threshold level will stop the shrinking of circles on an other layer.

Figure 4.23 Radii Map generation from the Distance Map - I.

The next Figure shows steps of Radii Map generation for picture containing two connecting circles.



Distance Map | Smoothed by GlobMax | Intensities at center points | Radii Map with GlobMax

Figure 4.24 Radii Map generation from Distance Map - II.

4.2.2.2 Model Generation

We use wave propagation phenomena to generate “dynamic bubble models” as follows. The picture containing center points of bubble groups are the initials of trigger waves. The waves will spread from these points producing larger and larger circles around those center points. Knowing that these set unions of circles contain different circles in size, it is necessary to constrain the propagation of the trigger wave. A spatio-temporal control will allow this restricted propagation to take place. This control image should be constructed in such a way that it contains information related to radii of objects. Thus the propagation of the trigger waves can be stopped at proper time. The Radii Map will be used as control for this task.

Figure 4.25 shows the process of generating dynamic bubble models. The initial points for trigger waves are the Center Points. Without any constraint the waves would propagate through all the cells of the network. The gradually smoothed Radii Map will control the spread of trigger waves enabling the propagation only at those positions where cell values have positive levels. If the pixels around center points reach the zero level the propagation of trigger waves will be stopped.

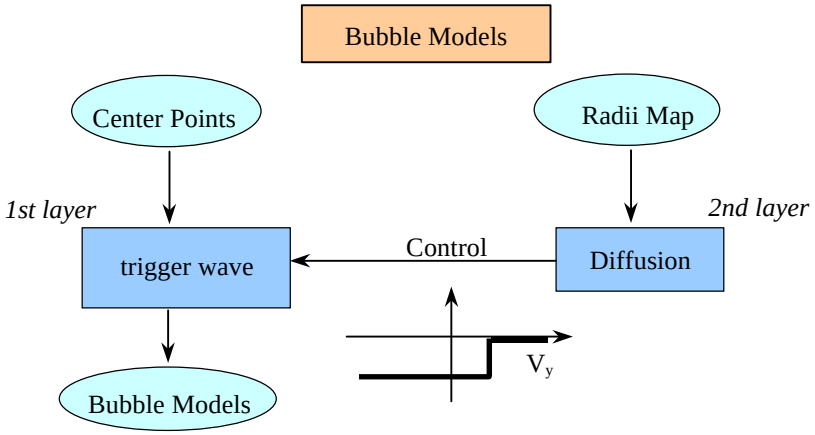


Figure 4.25 Generating bubble models. The center points of the objects are initial points of the trigger waves. To stop the spreading of waves a spatio-temporal control is applied – Radii Map. This control image contains information related to radii of objects. Where cell values of the control are larger than a given level then the spreads of trigger waves are enabled. The diffusion on the Radii Map results in a gradually smoothing. When cell values decrease below a given level then trigger waves will be stopped.

The center points and radii map are used to generate bubble models in such a way that bubble models should be perfect circles or groups of circles. Figure 4.26 shows test pattern for tuning bubble model generation. Template values were computed through an optimization process based on Hamming distance minimization. The non-linearity causes that it is impossible to minimize error for each size of circles. The optimum was set to the class of most frequently occurring bubble sizes.

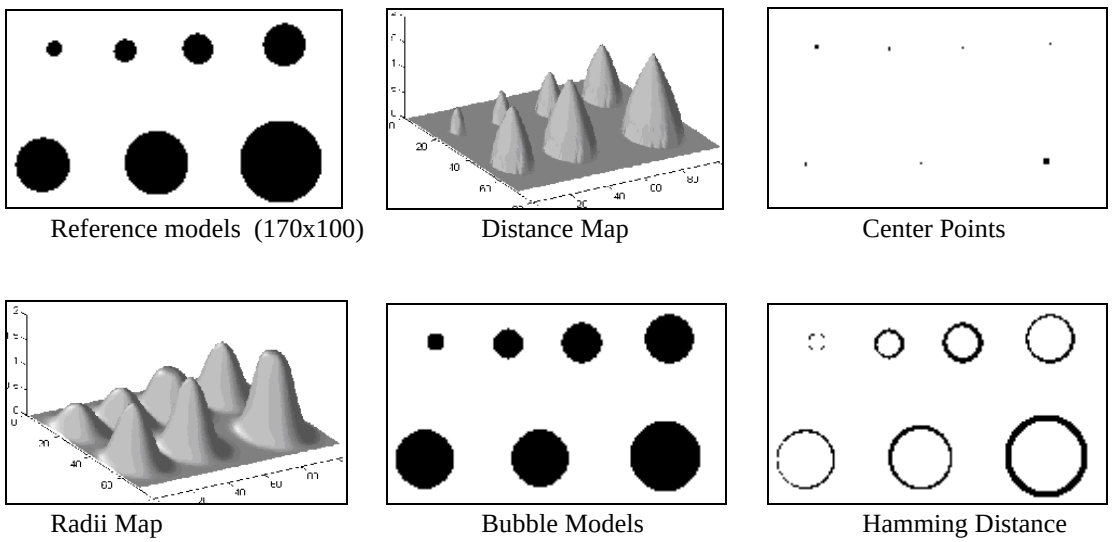
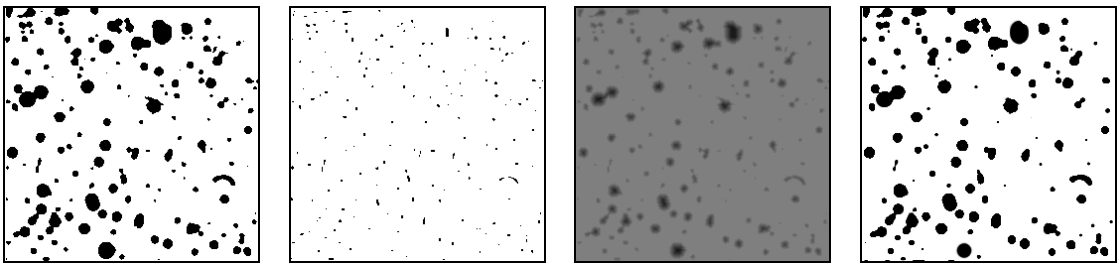


Figure 4.26 Test image for tuning bubble model generation. Optimization algorithm finds template values resulting lowest Hamming distance among input objects and bubble models.

Because Hamming distances cannot be optimized for each size of circles the distance measure for classification should be an other method. In *Chapter 3* we saw that for this type of problem the nonlinear Hausdorff distance is a good candidate because object's extensions are penalized stronger then differences.

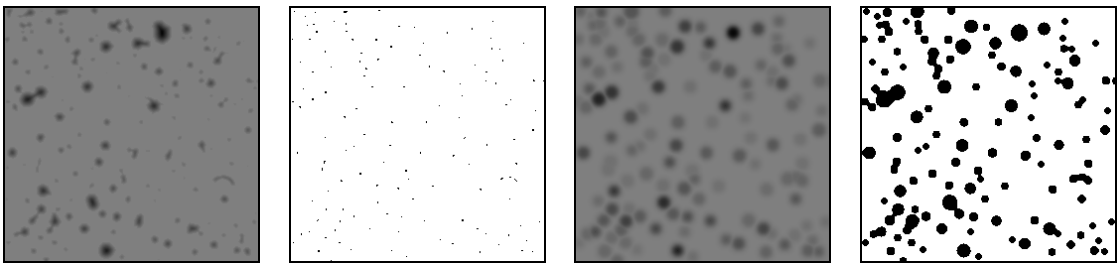
4.2.2.2.1 Comparison of two methods in bubble models generation

I. Center points are generated via masked erosion. The smoothed Euclidean Distance Map is used as Radii Map.



input (256x256) Center Points - I Distance Map as Radii Map Bubble Models (1)

II. Center points are generated via Distance Map. The Radii Map is constructed from the Distance Map.



Distance Map Center Points - II Radii Map Bubble Models (2)

Comparison between different model generation shown in Table 4.2

Bubble Model Generation	Advantages	Disadvantages
Iterative Morphology	• weak parameter sensitivity	• models are not groups of circle
Distance Map	• models are circles	• parameter sensitivity

Table 4.2 Comparison of different methods for bubble model generation.

Finally the concrete training set was contained fifty real images like in Figure 4.27.

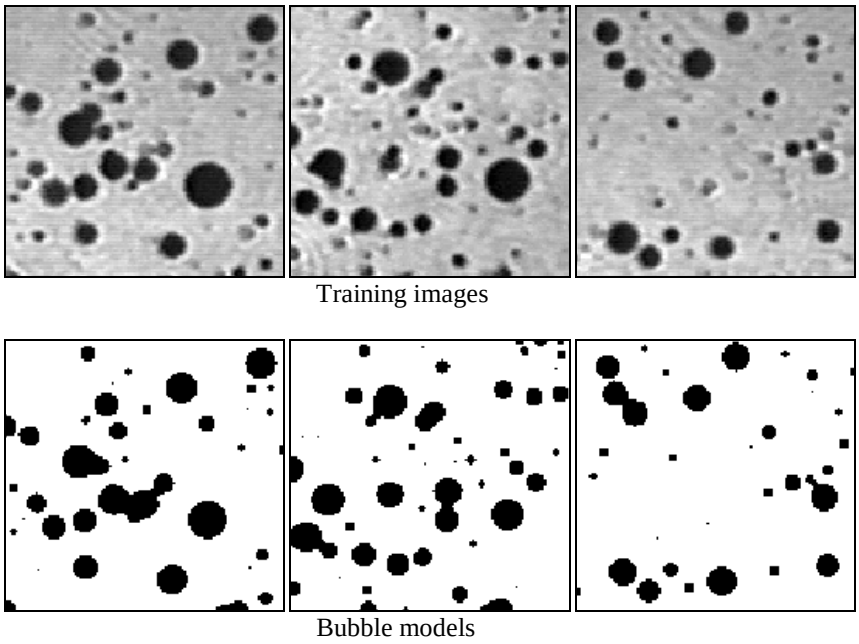


Figure 4.27 Elements of training set for bubble model generation. Fifty images with size 128x128 were used. Total amount of objects was around 2500-2600.

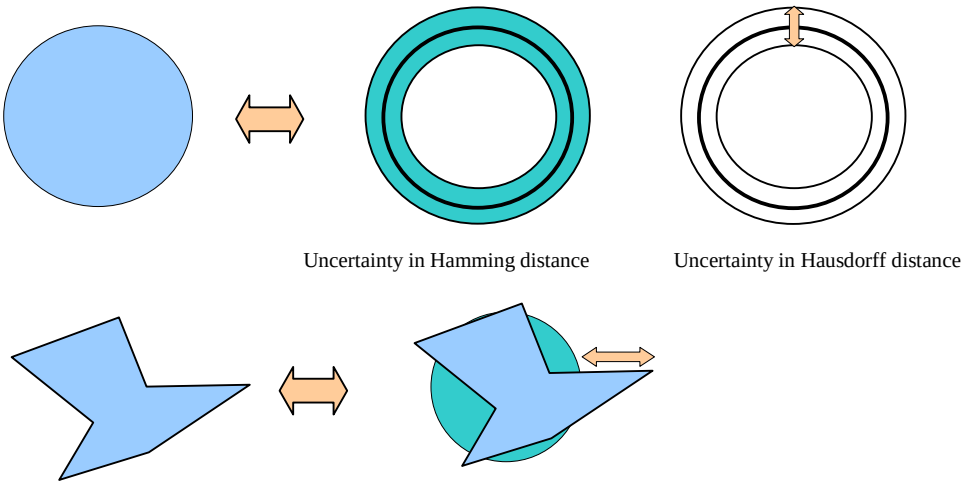
4.2.2.3 Classification Based on Nonlinear Hausdorff Metric

The reason for proposing the nonlinear Hausdorff metric to solve this classification problem is the following. Relying on nonlinear Hausdorff distance calculation a qualitatively better solutions can be derived compared to different variants of the Hamming metrics. Furthermore, though it is computationally expensive, the CNN hardware ensures an efficient execution. In the CNN framework to compute the Hamming distance needed for classification the number of detected points should be normalized with reference to the object's area. This means that we cannot define a global threshold level and each object should be separately examined. The implemented nonlinear Hausdorff metric measures differences between objects and their models, converting the CNN transient length to a corresponding gray-scale level. In this case a global threshold level can be used for all objects in the classification step.

As it was mentioned before, during the model generation the uncertainty of the Hamming distance is larger than for the Hausdorff distance. This is illustrated in Figure 4.28. In case of a debris particle the Hamming distance between this particle and its bubble model might be less than the uncertainty among bubbles and their models. There are debris particles, which have quite long extensions and these extensions might be thin as well. This cannot be measured efficiently using the Hamming distance. The Hausdorff distance is especially adequate to measure these extensions. In Figure 3.17 and 3.18 were shown examples to demonstrate that Hausdorff distances penalizes for long extensions and not for many pixels that are near to the basic object. This property of the Hausdorff distance forms the base why it is chosen for the classification.

Another possibility would be to put the question of the metric choice into a decision theoretical context using statistical framework. But it is very hard to state anything about the underlying distribution because the complexity of the problem and the multiple-stage algorithm with many non-linear interactions render this question nearly impossible to answer. One

possibility is to investigate the model generation itself. The optimization method computes at each step for many bubble-objects their models and the sum of the Hamming distance is calculated. This cumulative error is minimized during the optimization process. The central limit theorem states that under very general conditions, the sum of random variables approaches the distribution of a Gaussian random variable. If we consider the single Hamming distance as a random variable then this suggests the hypothesis that models of bubbles will vary depending on a Gaussian distribution.



In case of a debris particle the Hamming distance might be less than the uncertainty of the Hamming distance for bubbles. This causes misclassification. But this uncertainty is much less using the Hausdorff distance.

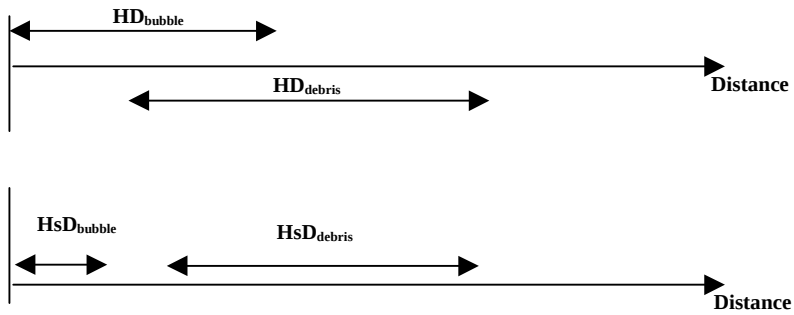


Figure 4.28 Uncertainty of model generation causes uncertainty in distance calculation based classification. The choice of the Hausdorff distance computation makes possible to minimize this error.

This implies that in the classification stage the uncertainty of the different metrics will depend on a Gaussian distribution. This can be modeled if we compute metrics for a given object and its model and the model is perturbed by a specific noise depending on a Gaussian distribution. The most sensitive component in the model generation is the size of the bubble models. This can be modeled that the model scale is varying. Such investigations were carried out in *Chapter 3.4* where properties of the Weighted Hamming metric were discussed (e.g. Figure 3.35). Figure 4.29 shows the measurement of the variations of metrics depending on a Gaussian distribution. The perturbation model was to scale bubble models. As it can be seen the Hausdorff distance shows significant difference between distributions depending on that the reference object is a bubble or a debris particle. The distributions of the Hamming distance are overlapping strongly while the distributions of the Hausdorff distance are much further from

each other. The Weighted Hamming shows also different distributions but it is less significant than in case of the Hausdorff distance. Therefore the Hausdorff distance is chosen to measure differences between objects and their models.

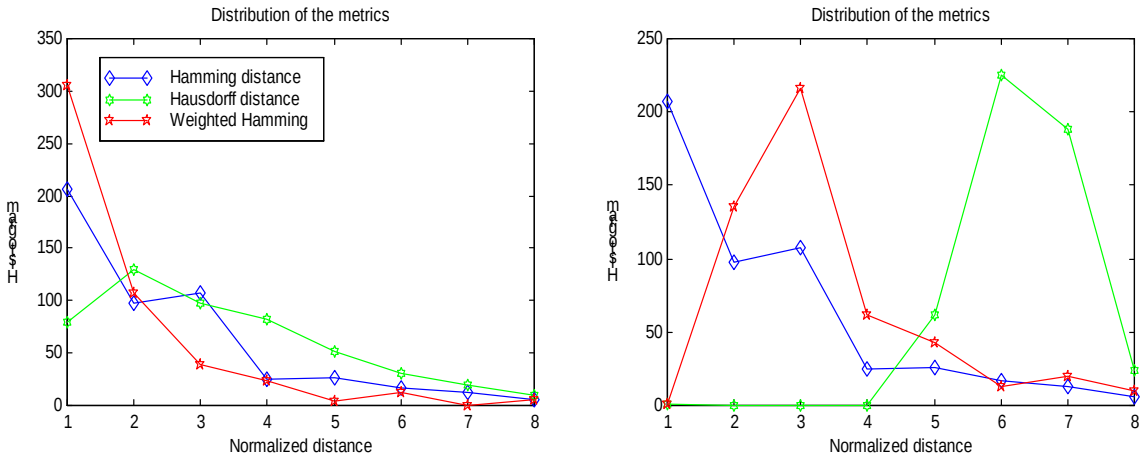


Figure 4.29 The uncertainty of metrics depending on the uncertainty of the model generation. This was modeled as the size of the bubble models depends on a Gaussian distribution. a) uncertainty for bubbles, b) uncertainty for debris particles.

Next, some results will be presented applying the classification algorithm to real images. Figure 4.30 shows consecutive steps of bubble-debris classification algorithm. The gray-scale input picture contains three debris particles the remaining objects are all bubbles. Although some bubbles are connected to each other the algorithm can recognize that these are not debris particles. However, further examinations are necessary to tune the algorithm to achieve the desired robustness in order to implement it on a real CNN chip and fulfill the requirement of the very low false alarm rate.

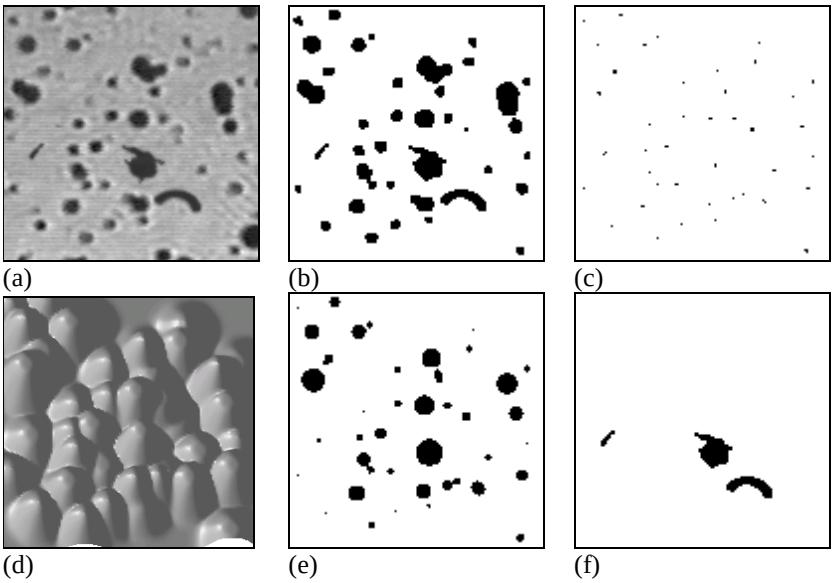


Figure 4.30 Consecutive steps of the bubble-debris classification algorithm on a real example. (a) original gray-scale image, (b) adaptively threshold, (c) center points, (d) radii map, (e) bubble models, (f) detected debris particles

We have trained and tested the performance of the algorithm on two different image sequences containing fifty images, respectively. The size of the images was 128x128. The total number of objects was between 2500-2600 and there were 130-150 debris objects. Each image contained approximately 40-50 objects with 2-3 debris particles. Empirically, the 10% of the samples were enough to determine the parameters of the algorithm. Figure 4.31 shows some examples.

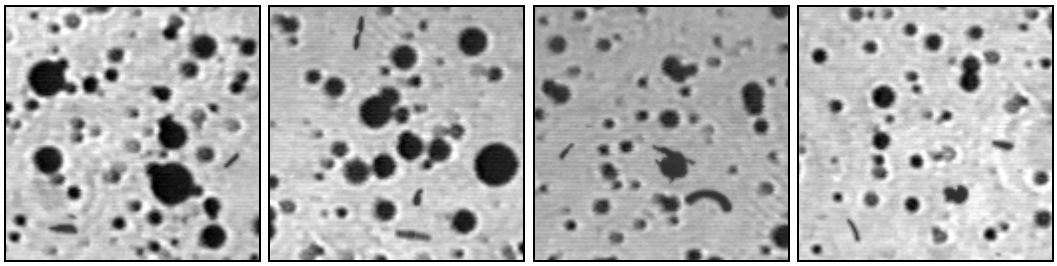


Figure 4.31 Gray-scale images containing bubbles with debris particles.

The power of the proposed algorithm was analyzed by constructing ROC (Response Operating Characteristic) shown in Figure 4.32. The false alarm rate was computed as the ratio of false classifications and the total number of objects. The true positive values were constructed as the ratio of detected debris particles and the total number of debris particles. Although the ratio of the classification of debris particles is relatively high and the false alarm rate is low, further improvement is still needed.

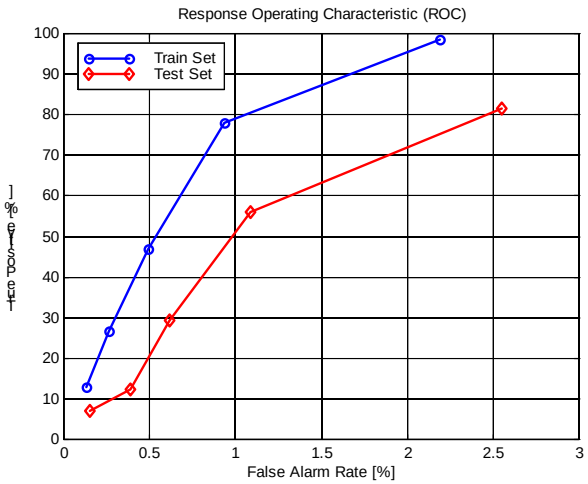


Figure 4.32 Response Operating Characteristic (ROC) of the proposed algorithm. The false alarm rate was computed as the ratio of false classifications and the total number of objects. The true positive values were constructed as the ratio of detected debris particles and the total number of debris particles.

4.2.3 Hardware Requirements of a Real time Application

In this part we address important implementation issues and analyze the expected time performance of the algorithm in a CNN Universal Chip environment and show the VLSI implementation complexity of different solutions.

4.2.3.1 Template robustness and stability

In template design the main consideration was to solve each template operation with the smallest neighborhood size (r = 1) since up to now this radius size is feasible only in CNN-UM chips. Ordinary templates in the algorithm (like threshold, average, etc.) are completely stable spatial operators. Also chip implementation experiments show their robustness in a real physical implementation.

Other important issue is the robustness of propagating type templates. Both simulation and chip experiments indicate that they can be implemented with proper accuracy, however two conditions should be satisfied that are slightly in contradiction with each other. The term *slightly* refers the fact that fortunately there is a common interval and the problem is tractable. The first condition is to provide as possible object's width independent wave propagation. It cannot be granted theoretically, however with faster wave propagation it can be approached. But it contradicts to the second condition that feature extraction, model production and time measurement via wave map generation have more accuracy if they are using slower wave propagation (i.e. more levels of quantization can be provided). As a compromise an intermediate setting was used.

The iterative type implementation requires no more specific features but the dynamic type needs two-layer nonlinear templates. The nonlinear interaction is the simplest, namely, central element of the template is used to control the filling of a layer for time measurement. The architectural details of a CNN-UM chip under development that can provide these requirements can be found in [27].

4.2.3.2 VLSI Implementation and Computational Complexity

Two types of the analogic algorithm for bubble-debris classification were presented. The iterative type implementation was designed especially in order to fit present CNN-UM chips and the simplified version of the algorithm was implemented on the 64x64 CNN-UM chip [12], [102]. It uses linear template operations, fixed-state map technique. This served only to test parts of the algorithm and to show their potentialities because the resolution of the chip and the time requirement of subroutines do not qualify it for a real time application (Table 4.3).

Task		**Iterative type implementation** (on the 64x64 CNN-UM chip)	**Dynamic type method**
Feature Extraction	*Adaptive filtering and threshold*	5τ+50τ	5τ+50τ
	Distance Map	-	5τ+20τ
	Center Points	5τ+50*(5τ+5τ)+5τ+15τ	5τ+12τ
	Radii Map		5τ+14τ
Bubble Models		5τ+50*(5τ+5τ)	5τ+15τ
Wave Metric	*Initial set generation*	5τ+5τ	5τ+5τ
	Wave Mapping	5τ+50*(5τ+5τ)	5τ+30τ
	Distance calculation (nonlinear Hausdorff)	5τ+5τ	5τ+5τ
Estimated running time in τ CNN		1610τ	191τ
Estimated running time (τ =100-200 nsec in present CNN-UMs) here τ = 200 nsec		= 322 μs	< 40 μs

Table 4.3 Running time specification for operations in bubble-debris classification algorithm. The timing is given in time constant measure (τ = RC). The 5τ in each subroutine stands for the time necessary for reprogramming. Even in the worst case the dynamic type method completes the classification for a single frame in less than 40 μs taking into the consideration the characteristics of the presently available VLSI implementation of a CNN core cell. Although the iterative type method requires also less than 1 msec but due to the chip resolution additional image partitioning would be necessary. Requirement is the real time processing at frame rate 1000 frame/sec with image resolution 512x512.

The main advantage of the dynamic type implementation is its very fast computability. This is the major requirement for a real-time application. As it was mentioned before, in the bubble-debris classification project the highest frame rate can be 1000 frame/sec with image size 512x512. For each step of the algorithm (Feature Extraction, Bubble Model generation, classification based on nonlinear Hausdorff metric) the same architecture with similar type of operation is used therefore reprogramming does not require additional template and LAM/LLM transfers. The important thing to address is the resolution of the CNN Universal chip. At least a 128x128 processing array is required (256x256 image sub-window can be reduced by direct averaging of the neighboring pixels without significantly effecting output quality). With this resolution sub-window processing and merging fits still into the time requirement. The 128x128 array size (with all required functionality) seems reasonable in the near future using 0.35-0.5 μm technology. It should be emphasized that the estimated running time is independent of the CNN-UM chip resolution although global propagation (trigger wave technique) is used. But they are designed for the occurring largest object's size therefore propagation time can be restricted.

4.2.4 Feasibility of Other Types of Solutions

The main requirements of an efficient solution are the following. The image frame rate can be as high as 1000 frames/sec. The image size is 512x512. In an image the number of objects is approximately 700-800. This means that in 1 ms the total inspection of an image around with 800 objects should be completed. These strong requirements make impossible to solve this task

in a sequential processing. Segmentation, labeling and analyzing objects separately would require enormous computing power. Therefore any solution based on a learning algorithm cannot be used where investigations are carried out on objects after each other. It means that image scene should be globally analyzed. Spectral domain analysis, histogram calculation, and correlation based techniques could be candidates. For instance, to use correlation technique efficiently at least 8-10 different base circle shapes would be necessary to filter out large and small bubbles jointly. Otherwise the threshold value cannot be set to a proper level to filter out bubbles but not debris or do not generate false alarm in case of bubbles as well. On the other hand, the resolution of images is not so fine that difference among bubbles and debris particles would significantly appear employing these global techniques. Somehow the geometrical properties at local level should be investigated. These two contradicting requirements can be done if the investigation carries out the total process parallel for each object (global processing) and these inspections take place at the low level of resolution. That is fulfilled exactly with the proposed solution.

4.3 Conclusions

I have discussed theoretical consideration of model based object classification using dynamical type operators especially focusing to the so-called non-equilibrium algorithms. Its importance is given by the revolution of the sensor technique what we have been facing for this decade. Especially for array type analog signals there is a requirement for high performance devices that are capable to process information near to the sensors. Obviously, in case of real time applications it seems to be the best solution if sensing and processing takes place in the same device avoiding unnecessary data conversion and transfer. For this novel type of devices we need different problem handling and algorithmic solutions.

I have outlined a possible framework for spatio-temporal classification algorithms in which spatio-temporal controlled nonlinear dynamical processes extract features of objects, generate their models, and classify them via the nonlinear wave metric computation. This framework exploits the advantages of the parallel computation and the advantages of the spatio-temporal dynamical processes in which transients of operations consist of fast and efficient algorithms for real time applications.

I have presented a possible solution for the bubble-debris classification problem, where the major requirement is real-time processing with a very low false alarm rate for miss-classified bubbles. The discussed analogic CNN algorithm combines different constrained wave propagation steps in order to extract features of objects, generate models and compute wave based metric among objects and models. The effectiveness of the algorithm was tested on sequence of real images and the implementation possibilities and complexity was thoroughly analyzed. The initial experiments indicate that the CNN-UM architecture with the nonlinear Hausdorff metric computation can distinguish bubbles and debris particles. Due to its high computing power it is capable for this high speed classification.

5 SUMMARY AND POTENTIAL APPLICATIONS

I have studied theoretical questions and problems related to object comparison and classification in the field of image processing, including the problems of metrics and distances, the possibilities of comparison and similarity measurement. In most cases the well-known distance computation techniques are used although it is a nontrivial problem what kind of metric is an optimal choice.

The approach presented in *Chapter 3* exploits the possibilities of spatio-temporal dynamical processes for object comparison and metric computation. One of the novel types of metrics (Weighted Hamming) includes the already known distance computations (Hamming and Hausdorff) as special cases. These wave mapping based metrics extend the space of the measurable properties of the objects with dynamical information. Geometrical properties of objects are extracted via different types of propagating waves and time evolution of this propagation encodes the information about similarity and difference. Examples show many advantages of these metrics and demonstrate that they can be used efficiently for object classification. The generalized wave mapping based metric computation methodology for 2D object shape comparison demonstrates that use of spatio-temporal dynamical processes extend the possibilities of this research area and several ways are opened for new type of metrics.

To compute these dynamical features efficiently a tool is required that makes it possible that through a sophisticated spatio-temporal dynamics objects are explored and their properties are extracted. The analogical CNN Universal Machine architecture seems to be an ideal candidate for this type of task and I have also presented some results on the present CNN-UM chips.

In the second part of my work (*Chapter 4*) I have discussed theoretical consideration of model based object classification using spatio-temporal dynamical type operators especially focusing to the so-called non-equilibrium algorithms. I have outlined a possible framework for spatio-temporal classification algorithms in which spatio-temporal controlled nonlinear dynamical processes extract features of objects, generate their models, and classify them via the nonlinear wave metric computation. This framework exploits the advantages of the parallel computation and the advantages of the nonlinear dynamical processes in which transients of operations serve as elementary operators for spatio-temporal algorithms. These algorithms are especially well suited for real time applications in which data flow should be processed at very high speed.

Important applications can use this type of measurement for comparisons and classifications. One of them could be the presented problem in *Chapter 4.2* where the major requirement is the real-time processing with a very low false alarm rate for miss-classified bubbles. The discussed analogic CNN algorithm combines different constrained wave propagation steps in order to extract features of objects, generate models and compute wave based metric among objects and models. Here other sub-results can also be used.

Besides, this part of my work demonstrates how spatio-temporal dynamics can be used to solve difficult tasks, where the requirement of real time analysis and the enormous volume of data to be processed cannot be fulfilled with traditional sequential computers.

6 REFERENCES

6.1 Publications on CNN Technology

[1] L. O. Chua and L. Yang, "Cellular Neural Networks: Theory", IEEE Trans. on Circuits and Systems, Vol. 35, pp. 1257-1272, October 1988.

[2] L. O. Chua and L. Yang, "Cellular Neural Networks: Applications", IEEE Trans. on Circuits and Systems, Vol. 35, pp. 1273-1290, October 1988.

[3] L. O. Chua, and T. Roska, "The CNN Paradigm", IEEE Trans. on Circuits and Systems, Vol. 40, pp.147-156, March 1993.

[4] T. Roska and L. O. Chua, "The CNN Universal Machine: an Analogic Array Computer", IEEE Trans. on Circuits and Systems, Vol. 40, pp. 163-173, March 1993.

[5] L. O. Chua, "CNN - Vision of Complexity", International Journal of Bifurcation and Chaos, Vol. 7, No. 10, pp. 2219-2425, October 1997.

[6] 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, April 1993.

[7] T. Roska, "Analogic Computing: System aspects of Analogic CNN Sensor Computers", 2000 6th IEEE International Workshop on Cellular Neural networks and their Applications Proceedings, pp. 73-78, Catania, Italy, May 23-25, 2000.

[8] T. Roska and L. O. Chua, "On a Framework of Complexity of Computations on Flows Implemented on the CNN Universal Machine", Technical Report, DNS-15-1995, Analogical and Neural Computing Laboratory, Computer and Automation Research Institute, Budapest, 1995.

[9] T. Roska, and L. O. Chua, "Computer-Sensors: Spatial-Temporal Computers for Analog Array Signals, Dynamically Integrated with Sensors", Journal of VLSI Signal Processing Systems, Vol. 23, pp. 221-238, 1999.

[10] J. M. Cruz and L.O. Chua, "A 16x16 Cellular Neural Network Universal Chip: the First Complete Single-chip Dynamic Computer Array with Distributed Memory and with Gray-scale Input-output", Analog Integrated Circuits and Signal Processing, Vol. 15. No. 3. pp. 227-238, 1998.

[11] S. Espejo, R. Carmona, R. Domingúez-Castro, and A. Rodrigúez-Vázquez, "CNN Universal Chip in CMOS Technology", International Journal of Circuit Theory and Applications, Special Issue on CNN II: Part I, Vol. 24, pp. 93-111, 1996.

[12] S. Espejo, R. Domínguez-Castro, G. Liñán, Á. Rodríguez-Vázquez, "A 64×64 CNN Universal Chip with Analog and Digital I/O", in Proc. 5th Int. Conf. on Electronics, Circuits and Systems (ICECS'98), pp. 203-206, Lisbon, September 1998.

[13] A. Paasio, A. Kananen, K. Halonen and V. Porra, "A 48 by 48 CNN Chip Operating with B/W Images", in Proc. 5th Int. Conf. on Electronics, Circuits and System (ICECS'98), pp.191-194, Lisbon, September 1998.

[14] Á. Zarándy, P. Keresztes, T. Roska and P. Szolgay, "CASTLE: an Emulated Digital CNN Architecture; Design Issues, New Results", in Proc. 5th Int. Conf. on Electronics, Circuits and Systems (ICECS'98), pp.199-202, Lisbon, September 1998.

[15] F. Werblin, T. Roska, and L. O. Chua, "The Analogic CNN Universal Machine as a Bionic Eye", International Journal of Circuit Theory and Applications, Vol. 23, pp. 541-569, 1995.

[16] P. Thiran, K. R. Crounse, L. O. Chua, and M. Hasler, "Pattern Formation Properties of Autonomous Cellular Neural Networks", IEEE Trans. on Circuits and Systems, Vol. 42, pp. 757-774, 1995.

[17] K. R. Crounse, L. O. Chua, P. Thiran, and G. Setti, "Characterization and Dynamics of Pattern Formation in Cellular Neural Networks", International Journal of Bifurcation and Chaos, Vol. 6, pp. 1703-1724, 1996.

[18] K. R. Crounse and L. O. Chua, "Methods for Image Processing and Pattern Formation in Cellular Neural Networks", IEEE Trans. on Circuits and Systems, Vol. 42, No. 10, pp. 583-601, 1995.

[19] P. Arena, S. Baglio, L. Fortuna, and G. Manganaro, "Self-Organization in a Two Layer CNN", IEEE Trans. on Circuits and Systems, Vol. 45, No. 2, pp. 157-162, 1998.

[20] P. Arena, L. Fortuna, M. Branciforte, "Reaction-diffusion CNN Algorithms to Generate Artificial Locomotion", IEEE Trans. on Circuits and Systems, Vol. 46, No. 2, pp. 253-260, 1999.

[21] Cs. Rekeczky and L. O. Chua, "Computing with Front Propagation: Active Contour and Skeleton Models in Continuous-time CNN", Journal of VLSI Signal Processing Systems, Vol. 23, No. 2/3, pp. 373-402, November-December 1999.

[22] L. O. Chua, M. Hasler, G. S. Moschytz, and J. Neirynck, Autonomous Cellular Neural Networks: A Unified Paradigm for Pattern Formation and Active Wave Propagation, IEEE Trans. Circuits Syst., Part I, Vol. 42, No. 10, pp. 559-577, 1995.

[23] V. Perez-Munzuri, V. Perez-Villar, and L. O. Chua, Autowaves for image processing on a two-dimensional CNN array of excitable nonlinear circuits: flat and wrinkled labyrinths, IEEE Trans. Circuits Syst., Vol. 40, pp. 174-181, Mar. 1993.

[24] 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. Circuits Syst., Part I, Vol. 42, No. 10, pp. 807-815, 1995.

[25] T. Kozek, L. O. Chua, T. Roska, D. Wolf, R. Tetzlaff, F. Puffer and K. Lotz, Simulating Nonlinear Waves and Partial Differential Equations via CNN-Part II: Typical Examples, IEEE Trans. Circuits Syst., Part I, Vol. 42, No. 10, pp. 816-820, 1995.

[26] Csaba Rekeczky, "Dynamic Spatio-temporal Nonlinear Filtering and Detection on CNN Architecture – Theory, Modeling and Applications", PhD dissertation, Analogical and Neural Computing Laboratory, Computer and Automation Institute of Hungarian Academy of Sciences, 1998.

[27] Csaba Rekeczky, T. Serrano-Gotarredona, Tamás Roska, and Ángel Rodríguez-Vázquez, "A Stored Program 2nd Order/3-Layer Complex Cell CNN-UM", Proceeding of the 6th IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA 2000), pp. 213-217, May 23-25, 2000.

[28] Á. Zarándy, A. Stoffels, T. Roska, F. Werblin, and L. O. Chua, Implementation of Binary and Gray-Scale Mathematical Morphology on the CNN Universal Machine, Memo No. UCB-ERL, Univ. of Cal. Berkeley, 96/19, 1996.

[29] Á. Zarándy, A. Stoffels, T. Roska, and L. O. Chua, "Morphological Operators on the CNN Universal Machine", CNNA'96, Fourth IEEE International Workshop on Cellular Neural Networks and their Application, pp. 151-156, Seville, Spain, June 24-26, 1996.

[30] T. Kozek, K. R. Crounse, T. Roska, and L. O. Chua, "Multi-Scale Image Analysis on the CNN Universal Machine", CNNA'96, Fourth IEEE International Workshop on Cellular Neural Networks and their Application, pp. 69-74, Seville, Spain, June 24-26, 1996.

[31] Á. Zarándy, "The Art of CNN Template Design", International Journal of Circuit Theory and Applications, Vol. 27, pp. 5-23, 1999.

[32] "CSL – CNN Software Library (Templates and Algorithms)", v. 7.1, Analogical and Neural Computing Laboratory, Computer and Automation Institute of Hungarian Academy of Sciences, 1997.

6.2 Publications on Related Research Fields

General topics

[33] László Lovász and Michael D. Plummer, "Matching theory", Akadémiai Kiadó, Budapest, 1986.

[34] J. Dugundji, "Topology", Allyn and Bacon, Inc., Boston, 1966.

[35] John C. Russ, "The Image Processing Handbook", CRC Press, 1995.

[36] Bernd Jahne, "Digital Image Processing", Springer-Verlag, 1991.

[37] A. Jain, "Fundamentals of Digital Image Processing", Prentice-Hall, 1989.

[38] D. Ballard and C. Brown, “Computer Vision”, Prentice-Hall, 1982.
[39] L. Blum, F. Cucker, M. Shub, and S. Smale, “Complexity and real computation”, Springer, New York, 1998.
[40] Alberto Leon-Garcia, Probability and Random Processes for Electrical Engineering, Addison-Wesley Publishing Company, 1994.
[41] Oliver, B.M. – Cage, I.M, ”Electrical Measurements and Instrumentation”, New York, Mc Graw-Hill, 1971.
[42] Richard C. Dorf, “The Electrical Engineering Handbook”, CRC Press, 1993.
[43] Schnell László, ”Jelek és rendszerek méréstechnikája”, Budapest, Muszaki Könyvkiadó, 1985.
[44] Pataki Péter, ”Méréstechnika -1, -2”, Budapest, Tankönyvkiadó, 1990.

Metrics, distance transforms, and similarity measures

[45] Fred Attneave, “Dimensions of similarity”, American Journal of Psychology, Vol. 63, pp. 516-556, 1950.
[46] F. Gregory Ashby and Nancy A. Perrin, “Toward a unified theory of similarity and recognition”, Psychological Review, Vol. 95, No. 1, pp. 124-150, 1988.
[47] Teuvo Kohonen, “Self-Organization and Associative Memory”, Springer-Verlag, 1989.
[48] Teuvo Kohonen, “The Self-Organizing Map”, In Proceedings of the IEEE, Vol. 78, No. 9, pp. 1464-1480, 1990.
[49] L. L. Thurstone, “A law of comparative judgement”, Psychological Review, Vol. 34, pp. 273-286, 1927.
[50] Amos Tversky and Itamar Gati, “Studies of similarity”, In E. Rosch and B. Lloyds editors, Cognition and Categorization, Erlbaum, Hillsdale, N.J., 1978.
[51] Amos Tversky, “Features of similarity”, Psychological Review, Vol. 84, No. 4, pp. 327-352, July 1977.
[52] Amos Tversky and Itamar Gati, “Similarity, separability, and the triangle inequality”, Psychological Review, Vol. 89, pp. 123-154, 1982.
[53] Roger N. Shepard, “The analysis of proximities: Multidimensional scaling with unknown distance function”, Part I and Part II, Psychometrika, Vol. 27, pp. 125-140 and 219-246, 1962.
[54] Roger N. Shepard, “Toward a universal law of generalization for psychical science”, Science, Vol. 237, pp. 1317-1323, 1987.
[55] Carol L. Krumhansl, “Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density.” Psychological Review, Vol. 85, pp. 445-463, 1978.
[56] Simone Santini, Ramesh Jain, “Similarity Matching”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.?, No. ?, pp. ?, 1996.
[57] Simone Santini, Ramesh Jain, “Similarity Measures”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 9, pp. 871-883, September 1999.
[58] Joeseph O’Sullivan and Mar T. Keane, “Examining Similarity: Using Neural Network Techniques to Model Similarity Phenomena”, TCD-CS-92-36, October 1992.
[59] Bruce G. Batchelor, “Pattern Recognition: Ideas in Practice”, New York:Plenum Press, pp. 71-72, 1978.
[60] Edwin Diday, “Recent Progress in Distance and Similarity Measures in Pattern Recognition”, Second International Jooint Conference on Pattern Recognition, pp.534-539, 1974.
[61] Morton Nadler, and Eric P. Smith, “Pattern Recognition Engineering”, New-York:Wiley, pp.293-294, 1993.
[62] H. Alt, K. Mehlhorn, H. Wagener, E. Welzl, “Congruence, similarity, and symmetries of geometric objects”, Discrete Comput. Geom. Vol. 3, pp. 237-256, 1998.
[63] M. R. Anderberg, “Cluster Analysis for Applications”, Academic Press, New York, 1973.

[64] Gail A. Carpenter, and Stephen Grossberg, “A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine”, Computer Vision, Graphics, and Image Processing, Vol. 37, pp. 54-115, 1987.

[65] Philip D. Wasserman, “Advanced Methods in Neural Computing”, New-York, NY: Van Nostrand Reinhold, pp. 147-176, 1993.

[66] R. Hect-Nielsen, “Counterpropagation Networks”, Applied Optics, Vol. 26, No. 23, pp. 4979-4984, 1987.

[67] David W. Aha, Dennis Kibler, and Marc K. Albert, “Instance-Based Learning Algorithms”, Machine learning, Vol. 6, pp. 37-66, 1991.

[68] David W. Aha, “Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms”, International Journal of Man-Machine Studies, Vol. 36, pp. 267-287, 1992.

[69] Dietrich Wetterschereck, David W. Aha, and Takao Mohri, “A Review and Comparative Evaluation of Feature Weighting methods for Lazy Learning Algorithms”, Technical Report AIC-95-012, Washington, D.C.: Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence, 1995.

[70] Randall D. Wilson, and Tony R. Martinez, “The Potential of Prototype Styles of Generalization”, In Proceedings of the Sixth Australian Joint Conference on Artificial Intelligence (AI’93), pp. 356-361, 1993.

[71] B. K. Natarajan, “Machine Learning: A Theoretical Approach”, Morgan Kaufmann, San Mateo, CA, May 1991.

[72] S. Muggleton, “Inductive Acquisition of Expert Knowledge”, Addison Wesley, Reading, Massachusetts, 1990.

[73] Chris Atkeson, Andrew Moore, and Stefan Schaal, “Locally weighted learning”, to appear in Artificial Intelligence Review, 1996.

[74] K. Popper, “Some Comments on Truth and the Growth of Knowledge”, In E. Nagel, P. Suppes, and A. Tarski, editors, Logic, Methodology, and Philosophy of Science, pp. 285-292, 1962.

[75] K. Popper, “A Note on Verisimilitude”, British Journal for the Philosophy of Science, Vol. 27, pp. 147-159, 1976.

[76] G. Oddie, “Likeness to Truth”, D. Reidel Pub. Comp, Dordrecht, Holland, 1986.

[77] I. Niiniluoto, “Truthlikeness”, D. Reidel Pub. Comp, Dordrecht, Holland, 1987.

[78] D. Randall Wilson, and Tony R. Martinez, “Improved Heterogeneous Distance Functions”, Journal of Artificial Intelligence Research, Vol. 6, pp. 1-34, 1997.

[79] Thomas Eiter, and Heikki Mannila, “Distance measures for point sets and their computation”, Acta informatica, Vol. 34, pp. 109-133, 1997.

[80] D. P. Huttenlocher, G. A. Klanderman, W. Rucklidge, “Comparing Images Using the Hausdorff Distance”, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 15, No 9, pp. 850-863, September, 1993.

[81] D. Huttenlocher, K. Kedem, “Efficiently Computing the Hausdorff Distance for Point Sets under Translation”, in Proceedings of the Sixth ACM Symposium on Computational Geometry, pp. 340-349, 1990.

[82] A. Rosenfeld and J. Pfaltz, “Distance Functions in Digital Pictures”, Pattern Recognition, Vol. 1, pp33-61, 1968.

[83] Frederic Leymaire and Martin D. Levine, “Simulating the Grassfire Transform Using an Active contour Model”, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 14, No. 1, pp-56-75, Jan. 1992.

Emergence and dynamical processes

[84] H. Haken, “Synergetics”, Springer, Berlin, 1978.

[85] H. Haken, “Synergetics: From Pattern Formation to pattern Analysis and Pattern Recognition”, International Journal of Bifurcation and Chaos, Vol. 4, No. 5, pp. 1069-1083, 1994.

[86] V. Biktashev, V. Krinsky, H. Haken, “A wave approach to pattern recognition (with application to optical character recognition)”, Int. Journal of Bifurcation and Chaos, Vol. 4, No. 1, pp. 193-207, 1994.

[87] V. Krinsky, ed., “Self-Organization. Autowaves and Structures Far from Equilibrium. Synergetics”, Vol. 28, Springer, Berlin, 1984.

[88] V. Krinsky, V. Biktashev, and N. Efimov, “Autowaves principles for parallel image processing”, Physica D, Vol. 49, pp. 247-253, 1991.

[89] J. A. Sethian, “An Analysis of Flame Propagation”, Ph.D. Thesis, University of California Berkeley, 1982.

[90] J. A. Sethian, “Curvature and Evolution of Fronts”, Comm. in Mathematical Physics, Vol. 101, pp. 487-499, 1985.

[91] J. A. Sethian, “Numerical Algorithms for Propagating Interfaces: Hamilton-Jacobi Equationsand Conservation Laws”, Journal of Differential Geometry, Vol. 31, pp. 131-161, 1990.

[92] S. Osher, and J. A. Sethian, “Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulation”, Journal of Compatational Physics, Vol. 79, pp. 12-49, 1988.

[93] R. Malladi, J. A. Sethian, and B. C. Vemuri, “Shape Modeling via Front Propagation: a Level Set Approach”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 17, No. 2, 1995.

[94] R. Malladi and J. A. Sethian, “A Unified Approach to Noise Removal, Image Enhancement, and Shape Recovery”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 17, No. 2, 1995.

[95] A. J. Chorin, “Flame Advection and Propagation Algorithms”, Journal of Compatational Physics, Vol. 35, pp. 1-11, 1980.

[96] M. Kass, A. Witkin, D. Terzopoulos, “Snakes: Active Contour Models”, International Journal of Computer Vision, pp. 321-331, 1988.

[97] P. Perona and J. Malik, “Scale-Space and Edge Detection Using Anisotropic Diffusion”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, pp. 629-639, July 1990.

[98] L. Alvarez, F. Guichard, P. L. Lions, and J. M. Morel, “Axioms and Fundamental Equations if Image Processing”, Arch. Rational Mech. Anal, Vol. 123, pp. 199-257, 1993.

[99] J. Reintjes, R. Mahon, M. D. Duncan, L .L. Tankersley, A. Schultz, V.Chen, D. J. Kover, and P. L. Howard, “Optical Oil Debris Monitor”, Condition Monitoring ’94, pp 335-343, Swansea, U. K., March 21-25, 1994.

[100] J. Reintjes, R. Mahon, M. D. Duncan, L .L. Tankersley, A. Schultz, V.Chen, D. J. Kover, P. L. Howard, M. Chamberlain, S. Raghavan, and N. Gupta, “Optical Debris Monitoring”, Proceedings of the 49 th Meeting of the Society for Machinery Failure Prevention Technology, pp. 263-272, Virginia Beach, Virginia, April 18-20, 1995.

6.3 The Author's Publications

[101] **István Szatmári**, Abraham Schultz, Csaba Rekeczky, Tibor Kozek, Tamás Roska, and Leon O. Chua, "Morphology and Autowave Metric on CNN Applied to Bubble-Debris Classification", IEEE Trans. on Neural Networks, Vol. 11, No. 6, pp.1385-1393, November 2000.

[102] **István Szatmári**, Ákos Zarándy, Péter Földesy, and László Kék, "An Analogic CNN Engine Board with the 64x64 Analog I/O CNN-UM Chip", 2000 IEEE International Symposium on Circuits and Systems (ISCAS-2000), May 28 - May 31, 2000, Geneva, Switzerland

[103] **István Szatmári**, "The implementation of a Nonlinear Wave Metric for Image Analysis and Classification on the 64x64 I/O CNN-UM Chip", 6th IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA 2000), May 23-25, 2000, pp. 395-400, University of Catania, Italy

[104] Ákos Zarándy, Servando Espejo, Péter Földesy, László KéK, G. Linan, Csaba Rekeczky, Angel Rodriguez-Vázquez, Tamás Roska, **István Szatmári**, Tamás Szirányi, Péter Szolgay, "CNN Technology in Action", Proceedings of IEEE Int. Workshop on Cellular Neural Networks and Their Applications, (CNNA'2000), May 23-25, 2000, pp.79-81, University of Catania, Italy, ISBN 0-7803-6344-2

[105] Tamás Roska, Károly László, László Kék, Tibor Kozek, László Nemes, Csaba Rekeczky, **István Szatmári**, Ákos Zarándy, Sándor Zöld and Péter Szolgay, "A CNN Application Development Environment and Toolkit, CADETWin", Towards the Visual Microprocessor - VLSI Design and Use of Cellular Network Universal Machines, pp. 39-58, Chichester, Ed. by Tamás Roska and Angel Rodriguez-Vázquez, J.Wiley, 2000

[106] **István Szatmári**, Csaba Rekeczky, and Tamás Roska, "A Nonlinear Wave Metric and its CNN Implementation for Object Classification", Journal of VLSI Signal Processing, Special Issue: Spatiotemporal Signal Processing with Analogic CNN Visual Microprocessors, Vol.23. No.2/3. pp.437-448, November-December 1999.

[107] Péter Szolgay, Károly László, László Kék, Tibor Kozek, László Nemes, István Petrás, Csaba Rekeczky, **István Szatmári**, Ákos Zarándy, Sándor Zöld, and Tamás Roska, "The CADETWin Application Software Design System - A Tutorial", Proceedings of 14 European Conference on Circuit Theory and Design, Design Automation Day proceedings, (ECCTD 99-DAD), pp.58-68, Stresa, Italy, August 1999.

[108] Péter Szolgay, Ákos Zarándy, Sándor Zöld, Tamás Roska, Péter Földesy, László Kék, Tibor Kozek, Károly László, István Petrás, Csaba Rekeczky, **István Szatmári**, and Dávid Bálya, "The Computational Infrastructure for Cellular Visual Microprocessors", Seventh International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems (MicroNeuro99), pp. 54-60, Granada, 1999.

[109] **István Szatmári**, Tamás Roska, "The CNN implementation of wave type metric for image analysis and classification", IEEE Int. Workshop on Nonlinear Dynamics of Electronic Systems (NDES-98), pp. 137-140, Budapest, Hungary, July 16-18, 1998.

[110] Abraham Schultz, Csaba Rekeczky, **István Szatmári**, Tamás Roska, and Leon O. Chua, "Spatio-temporal CNN Algorithm for Object Segmentation and Object Recognition", 5th IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA-'98), pp. 347-352, London, April 14-19, 1998.

[111] **István Szatmári**, Abraham Schultz, Csaba Rekeczky, Tamás Roska, and Leon O. Chua, “Bubble-Debris Classification via Binary Morphology and Autowave Metric on CNN”, ERL-UCB Memo97/97, Electronics Research Laboratory, University of California at Berkeley, December 1997.

[112] Csaba Rekeczky, Abraham Schultz, **István Szatmári**, Tamás Roska, and Leon O. Chua, “Image Segmentation and Edge Detection via Constrained Diffusion and Adaptive Morphology: a CNN Approach to Bubble/Debris Image Enhancement”, ERL-UCB Memo97/96, Electronics Research Laboratory, University of California at Berkeley, December 1997.

[113] **István Szatmári**, Abraham Schultz, Csaba Rekeczky, Tamás Roska, and Leon O. Chua, “Bubble-Debris Classification via Binary Morphology and Autowave Metric on CNN”, 6th International Symposium on Nonlinear Theory and its Applications (NOLTA ’97), pp. 217-220, Hawaii, December 1997.

[114] Csaba Rekeczky, Abraham Schultz, **István Szatmári**, Tamás Roska, and Leon O. Chua, “Image Segmentation and Edge Detection via Constrained Diffusion and Adaptive Morphology: a CNN Approach to Bubble/Debris Image Enhancement”, 6th International Symposium on Nonlinear Theory and its Applications (NOLTA ’97), pp. 209-212, Hawaii, December 1997.

[115] Péter Szolgay, **István Szatmári**, and Károly László, "A Fast Fixed Point Learning Method to Implement Associative Memory on CNNs", IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications, Vol.44, No.4, pp. 362-366, April 1997.

[116] Tamás Roska, Péter Szolgay, Tibor Kozek, Ákos Zarándy, Csaba Rekeczky, László Nemes, László Kék, Károly László, **István Szatmári**, Márton Csapodi, "CADETWIN", CADETWIN, (CADETWIN), Budapest, MTA SZTAKI, 1997.

[117] Béla Fehér, Péter Szolgay, Tamás Roska, András Radványi, Tamás Szirányi, Márton Csapodi, Károly László, László Nemes, **István Szatmári**, Géza Tóth, and Péter Venetianer, “ACE: a Digital Floating Point CNN Emulator Engine”, IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA-’96), Seville, June 24-26, 1996.

[118] Károly László, **István Szatmári**, and Tamás Roska, “Implementation of the neocognitron on the CNN Universal Machine architecture”, Neuromorphic Information Technology, Graudate Center, NIT-3-1996, December 1996.

[119] Péter Szolgay, **István Szatmári**, and Károly László, “A Fast Learning Method to Implement Associative Memory on CNNs”, DNS-9-1996, Technical Report, Analogical and Neural Computing Laboratory, Computer and Automation Institute of the Hungarian Academy of Sciences, Budapest, November 1996.

[120] Péter Szolgay, **István Szatmári**, and Károly László, “A fast learning method to implement associative memory on CNN”, European Conference on Circuit Theory and Design (ECCTD ’95), pp. 991-994, Istanbul, August 1995.

[121] Péter Szolgay, Károly László, and **István Szatmári**, “A new learning algorithm to implement associative memory on CNN”, DNS-8-1995, Technical Report, Analogical and Neural Computing Laboratory, Computer and Automation Institute of the Hungarian Academy of Sciences, Budapest, December 1995.

7 APPENDICES

7.1 Theses of the dissertation

Methods used in the experiments

In the course of my work, theorems and assertions from the field of ordinary and partial differential equations, reported results on robustness and feasibility analysis connected to neural/nonlinear networks, methodologies related to image processing were explored.

Designed CNN templates and algorithmic procedures were tested on software simulators (tools of the CADETWIN package, [107,108]), on digital emulators implemented as hardware accelerator boards (HAB, ACE, [117]), and on a prototyping system embedding a VLSI CNN Universal Chip (CCPS, [102]). All these systems were developed in Analogical and Neural Computing Laboratory. The implementation independent description of the developed methods was completed in different "CNN languages" (CSD , Alpha) ensuring their applicability on different hardware platforms. During the design process the VLSI implementation complexity was minimized, therefore, for solving various tasks most templates applied required only nearest neighbor interactions. Most of these are linear interaction operators (templates) that can be tested on existing VLSI prototype systems or nonlinear operators with simple nonlinear interactions these operators are expected to be built in the near future. In software simulations, the numerical integration of coupled ordinary differential equations was based on first order explicit and implicit Euler formula with proper time step. In certain cases, where a higher precision was required, a higher order (fourth order Runge-Kutta) formula was also used.

New scientific results

1. Thesis: Nonlinear wave metric-class for 2D object comparison [101], [103], [106], [109], [111], [113]; *Chapter 3*

I have developed a methodology for 2D object comparison where geometrical (structural and morphological) properties of the object shapes are extracted by propagating trigger waves. The extracted information makes possible to compute several metrics for similarity and difference measurements.

These metrics measure both differences and similarities among objects. They inherently include dynamic type information on object's properties not only static features as in case of the widely used Hausdorff and Hamming distances. A dynamic type of information is, for instance, when the time evolution of the nonlinear waves during the comparison is also encountered. It is easy to give simple examples in which Hausdorff and Hamming metrics fail and they cannot distinguish certain objects. These drawbacks can be avoided by the nonlinear wave metric which includes these well-known distance measures (Hamming and Hausdorff) as special cases.

The demand of large amount of parallel computation and the requirement of the implementation of spatio-temporal dynamical processes can straightforwardly be mapped into parallel hardware architectures especially into analogical cellular computers. The CNN

architecture (Cellular Nonlinear/Neural Network) is dedicated to implement this nonlinear wave metric since it allows complex operations to be solved with a few elementary operations of the CNN and results in a fast and efficient computation. Except the spatial discretization all variables are analog (time, signals, interactions).

1.1 Autowave distance (nonlinear Hausdorff metric) computation on CNN architecture [101], [109], [111], [113]; *Chapter 3.3*

I have developed an efficient procedure to compute the Hausdorff metric on CNN architecture. The Hausdorff metric and its variant (autowave distance) which is less sensitive to noise effects were both implemented.

The solution is based on generating trigger waves propagating at an adaptive control and recording their propagation time at each special locations. In an actual CNN implementation the propagation time is transformed into charge capacitance of the CNN cells. The charge will be equivalent to the distance among positions.

Two types of implementation were presented. One of them operates on a two-layer CNN structure computing the autowave distance in one transient (one template operation - ~μs) while the other one is an iterative type implementation. The architecture of the first method is under design while the latter one can be tested on present CNN chips.

Taking VLSI complexity and accuracy constraints into account, I have specified the optimal trigger wave generation template group.

1.2 Introducing the Weighted Hamming (Integrated Hausdorff) metric and its implementation on CNN architecture [103], [106]; *Chapter 3.4, 3.5*

I have developed and defined the Weighted hamming (integrated Hausdorff) metric as a novel metric for object comparison.

Studying two well-known distance metrics (Hamming and Hausdorff metrics) and interpreting what they measure on objects, I showed their constraints and disadvantages. They reflect only static properties of objects like the distance of a point to another point or the sum of points in a set.

Using spatio-temporal dynamic processes to explore objects, the obtained information can provide much deeper examination. For instance, the Hausdorff and Hamming metrics can be combined by computing the sum of a wave map that is generated by trigger wave propagating on the objects. An intensity value at a given location encodes the time (and also the spatial distance) that is necessary for the front of the trigger wave to reach this location starting from an initial position (intersection of the two objects). In Figure 1 the generation of the wave map can be seen for two partially overlapping objects (objects are given as a set of points).

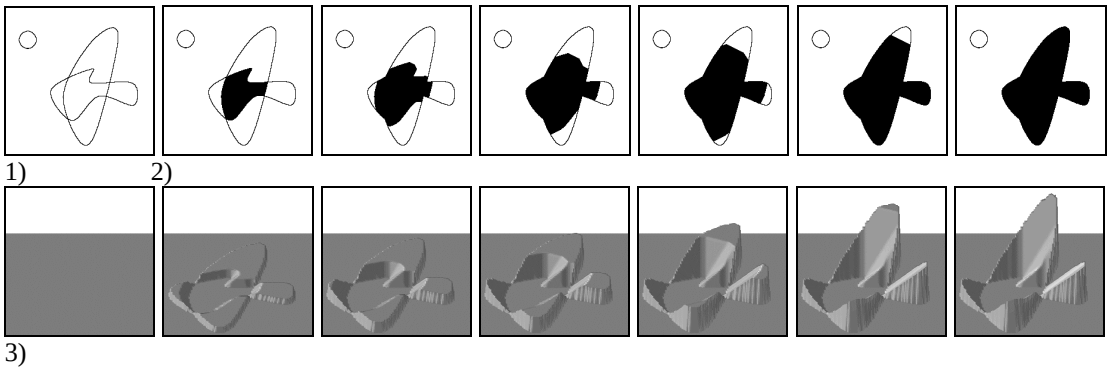


Figure 1 Wave map generation. 1) Outlines of two partially overlapping point sets. 2) Trigger wave spreads from the intersection through the union of contiguous part of point sets until all the points become triggered. 3) Wave map generated by increasing intensities of pixels until the trigger wave reaches them. This wave map contains encoded both the Hamming and Hausdorff distances.

Two explanations can be given for the sum of the wave map depending on the focus of attention.

- First, the sum of the wave map can be considered as the points of the Hamming distance are weighted in such a way that the intensity values of the wave map stand for these weights. Then these weighted Hamming points are summarized. The higher the intensity the farther the distance of the point is from the initial position.

- Another explanation might be that the intensities of the wave map stand for the "local" Hausdorff distances. Summing these values (in continuum this is the integration i.e. the volume of the wave map) gives the measure of the difference between two objects.

Both interpretations show that difference measurement relies on each point of the difference's set in which each point gives its specific value to the distance. Farther the point is from the intersection higher the weight with it contributes to the distance. It is opposite to the Hamming distance where each point has the same weight and it is also opposite to the Hausdorff distance where the distance of the farthest point results in the difference.

I have proved that the defined metric using this integration fulfills the conditions of being a metric. This metric incorporates the advantages both of the Hamming and the Hausdorff metrics.

This method computing the metric requires a complicated algorithm on digital computers, therefore, it would not have a practical application. Nevertheless, the analogic CNN architecture is an ideal tool to perform such kind of operations. The CNN based implementation I gave makes it possible to compute this metric very fast therefore it can be used in real application.

1.3 Generalization of the wave map based methodology for 2D object comparison and distance computation [106]; *Chapter 3.6*

I have introduced a nonlinear wave mapping based methodology for object comparison using spatial propagating trigger waves. The novelty of this approach is that objects are explored via spatio-temporal dynamic processes and their recorded progress in time gives the base of the comparison.

Over the objects to be compared trigger waves propagate and generate a special wave map. This wave map encodes the time evolution and dynamics of this propagation as an intensity map in such a way that the intensity value at a given position increases continuously until the trigger wave reaches that position. This wave map carries enough information that can be used for many different distance and comparison measurements. From this wave map several metrics can be computed depending on the chosen aspect, i.e. area integration, boundary scan, histogram analysis, skeleton based summarization, etc. The optimal choice depends on the type of the problem.

The nonlinear wave mapping based metric computation consists of three parts as follows.

$$\overset{I.}{D} = [\overset{II.}{W}, \overset{III.}{\{?, \mathrm{F}, \mathrm{G}, \mathrm{B}\}, d}]$$

The three parts corresponds to a three-step transformation where projections compress spatio-temporal information by reducing spatial and temporal dimensions into a single real number that gives finally the result of the metric computation. The D metric or distance has the steps as follows.

I. The *W(.)* is a spatio-temporal mapping generating a wave map.
II. The $\{?, \mathrm{F}, \mathrm{G}, \mathrm{B}\}$ intermediate processing computes a specific distribution or gray-scale or binary morphology computation.
III. The d identifies the final distance via a real function projection.

One subclass of this method was presented in the previous 1.2 thesis and includes the Hamming and Hausdorff distances as special cases.

The wave map makes it possible to encode the similarity of the objects in a single number instead of computing several features that would require many different computations. In classification tasks it might be useful to make decisions based on single number instead of a feature vector.

Based on a CNN-UM architecture implementation, several object pairs can be compared at the same time. Both the wave map construction and metric computation are computed very fast (~μs) that is necessary in real time applications.

2. Thesis: Use of spatio-temporal dynamics in algorithm design for model based object classification [101], [110], [111], [112], [113], [114]; *Chapter 4*

I have developed a model based classification algorithm based on spatio-temporal processes as elementary operators.

A novel approach to the solutions of object comparison problems relies on the use of the wide range of spatio-temporal dynamic processes. However, the associated operators might be so complex spatial and time interactions that their realizations by traditional digital computers might require enormous computer power. On the other hand, if these operators could be used as basic elementary operations then new kinds of algorithms could be developed.

The advantages of the metric presented in the first thesis could be effectively exploited if the algorithm is build up by similar type of operations. The proposed architecture for the nonlinear wave metric computation is appropriate to implement wide range of spatio-temporal dynamical processes. In the second thesis I will present an example of how a practical problem can be dealt with relying on a few number of spatial-temporal dynamic operators.

2.1 Spatio-temporal dynamical operators as elementary instructions in classification algorithms [114]; *Chapter 4.1*

I have developed a methodology for design of model based classification algorithms based on spatio-temporal operators.

There are several classification problems where large number of objects should be continuously classified or compared to their references in real time. It is important to solve these type of problems using only a few number of operations being *as simple as possible*. In these cases it is usually impossible to extract many features and combine them into a vector as well as to classify objects computing differences between these vectors. The only way is the use of the principle of *one measure – one decision*. For such kind of measurement the metric in the first thesis is especially effective because it encodes the static and dynamic properties in a simple spatial pattern. To perform this measurement a model (reference) of the objects is to be generated. Based on these considerations, an algorithm of this kind can generally be partitioned into three major parts as follows.

I. Compute some characteristically important features of the objects. For instance, the central points and the sizes of the objects might serve as primary data.
II. Generate models using both the previous and *a priori* information on the objects. An example for a priori information is the expected geometrical properties of the models.
III. Finally, the comparison and classification using the wave type metric.

Keeping in mind that the computation should be fast, I have chosen for all the three parts the same type of operators and architecture, namely, spatio-temporal controlled trigger wave propagation and recording the time evolution of this propagation. The identical types of

operators make it possible that a given physical implementation can have low complexity. The sub-results in each part of the algorithm are as follows.

I. a) I have developed a procedure to detect local center points of objects; where local means that if an object could be built up from geometric primitives (circle, rectangle, …) then these center points of primitives are the local center points of an object. This is useful to generate models of objects using these primitives. The basic idea of this procedure is that trigger waves shrink down the objects to single points. The pixel intensities corresponding to the objects are continuously growing in the second image until the trigger wave front reaches their positions. As a result the extracted information, called the wave map, contains local maxima and positions of these maxima encode the local center points.

   b) On the other hand the wave map can be viewed as the approximation of the information that can be obtained by the medial axis transformation. This can be used for other applications and the advantage of this implementation is that this operation can be executed on the CNN structure during a single transient using only one template operation. On digital computers this requires a quite sophisticated algorithm.

II. In the second part of the algorithm, trigger waves generate models with the help of a space-time variant control image. This is, in certain sense, the reverse process of the first part. The control image is obtained from the previous wave map. Its change and its interaction with the image where models are growing govern the propagation of trigger waves. This process establishes the final shapes of the models. The required geometrical properties of the models are encoded in the control image.

III. The classification is based on the wave metric presented in the first thesis. The type of the metric should be chosen depending on the given task.

These operators correspond to elementary operations of the analogic CNN-UM computer architecture therefore fast computation is easily achieved. Defining a two-layer structure, the core parts of the algorithm take only single transients. It results in a much higher speed (2-3 orders faster) than the time requirement would be on a sequential based hardware solution. I have also presented an iterative type of solution that can be tested on present CNN-UM chips.

2.2 Bubble-debris classification algorithm in condition based maintenance of engines using spatio-temporal dynamical operators on CNN architecture [101], [110], [111], [112], [113], [114]; *Chapter 4.2*

I have developed a real time procedure for solving the bubble-debris problem in condition based maintenance of engines using CNN architecture.

As a possible application of this type of algorithms I present a procedure to classify unknown objects scanned optically from the oil flow of the operating engines. This computation should be performed in real time. There are two basic classes of unknown objects: (i) the class of air bubbles and (ii) wear off particles. These debris particles might cause serious engine failure. Figure 2 shows a typical image that is already pre-processed (filtered and thresholded).

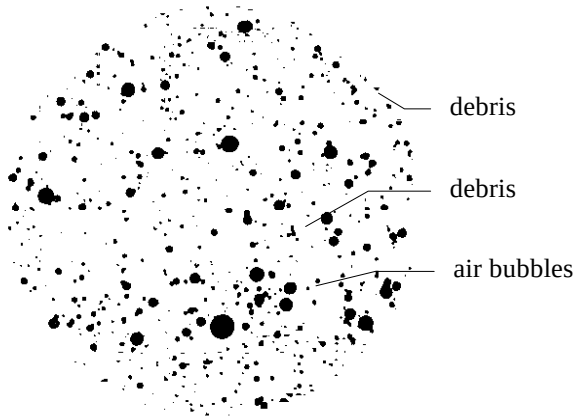


Figure 2 Typical image of the oil flow with floating particles as recorded by the CCD sensor. The image size is 512x512 and the pixel size is about 14μm. The frame rate can be as high as 1000 frames/sec.

The objective of this task is to construct a compact system that performs the inspection in real time and satisfies the requirement of an extremely "*low false alarm rate*". Images are to be processed in a very short time therefore a traditional solution based on sequential type hardware will not able to meet these requirements.

Due to these constraints I have developed an algorithm using the previous theoretical considerations and implemented it on the analogical CNN computer. The CNN type solution exploits the advantage that these types of procedures can be mapped to elementary CNN operations. Because the major part of the unknown objects belongs to the class of air bubbles, the models are circles or connected circle groups. The control image governing the growth of the bubble models encodes an isotropic trigger wave propagation. To achieve the *low false alarm rate* fine tuning is necessary. Based on the method in thesis 1.1, the wave map computation and distance measurements are performed during two transient times and the classification of the objects was successful. The chosen solution gives the possibility to solve this task in a very short time interval. The time requirement of the inspection is independent on the number of the unknown objects in an image because of the parallel computation. Based on experimental results I give time performance measurement of the algorithm on the present ACE4K chip and a theoretically ideal architecture that is under design.

Potential applications of the results

I have studied some basic questions and problems in the field of image processing, namely, the problem of metrics and distances, the possibility of comparisons and similarity measurements. These questions always arise if the task is related somehow to object classification. In most cases the well-known distance computation techniques are used although it is a nontrivial problem what kind of metric is an optimal choice.

The approach presented in the first thesis, combining two basic metrics, could be developed as a general measurement technique for object comparison. Besides, it includes the already

known distance computations as special cases and extends the space of the measurable properties of the objects with dynamic information.

To compute these features efficiently a tool is required that makes it possible that through a sophisticated space-time dynamics objects are explored and their properties are extracted. The CNN architecture seems to be an ideal candidate for this type of task and I have also presented some results on the present CNN-UM chip.

Important applications can use this type of measurement for comparisons and classifications. One of them could be the presented problem in the 2nd thesis. Here other sub-results can also be used. Besides, this part of my work demonstrates how spatio-temporal dynamics can be used to solve difficult tasks, where the requirement of real time analysis and the enormous volume of data to be processed cannot be fulfilled with traditional sequential computers.