Object Recognition From Colored Points Cloud Based on Artificial Neural Networks

Saeed Gholami Shahbandi

Master's thesis September 4, 2012
Automatique et Systèmes de Production
Robotics Group / IRCCYN
École Central de Nantes

Object Recognition From Colored Points Cloud Based on Artificial Neural Networks

Saeed Gholami Shahbandi

Master's Thesis in Robotics (30 ECTS credits)
IRCCYN - École Central de Nantes
LISA Laboratory - University of Angers
In framework of CART-O-MATIC project

Supervisors:
Philippe Lucidarme
Gaëtan Garcia

Jury Members:
Jean-François Lafay
Wisama Khalil
Marie-Françoise Lucas

Invited Member:
Michem Guglielmi

Abstract

An object recognition problem has been dealt within a mobile robot context. Robot's mission is environment exploration and object recognition for a competition (Dèfi CAROTTE). Visual Acquisition is carried out by an RGB-D camera embedded on the robot. Objects are isolated from the scene with respect to depth information. Four $3D$ global features vectors are proposed based on size and color of the objects, with help of PCA tool. These feature vectors are employed both independently and as a combined $12D$ feature vector. Two multi-class classifiers are proposed to handle the recognition problem. One classifier performs multi-labeling strategy and it's based on ANN and RBF networks. Along it's interesting modular mechanism independent from negative reinforcement, its performance is improved with a novel learning algorithm and compatible network structure. The other classifier conducts a single-labeling schema based on detecting frontier points of each class. Not only it is more precise compared to other classifiers such as SVM, Its simplicity makes it very attractive.

Key words:
Object Recognition, Classification, Multi-Class, Multi-Label, Artificial Neural Networks, RBF Networks, Feature Extraction, PCA, RGB-D Camera.

Dedicated to my parents

Zari & *Ali*

Acknowledgment

This work has been partially supported by the *French National Research Agency (ANR)* and *General Delegation for Armaments (DGA)* through the *Cart-O-matic* project in the *CAROTTE* challenge.

Part of this master's thesis has been published in *1st International conference on Systems and Computer Science (IEEE)*, Lille, France, 2012 [1]. Hence this reports shares some materials with mentioned publication.

With special thank to;

- Professor *Philippe Lucidarme* for all his helps and kindness, who dedicated a lot to this master's thesis. With his coordinations CARTOMATIC team reached the first place in CAROTTE competition,
- Dear Professors *Gaëtan Garcia*, *Nicolas Delanoue* and *Marie-Franoise Lucas* for their times and instruction,
- A dear colleague *Simon Landrault* for his helps,
- and all kind staff of LISA Laboratory.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

ANN	Artificial Neural Networks
HCI	Human Computer Interaction
HRI	Human Robot Interaction
HSV	Hue Saturation Value (color space)
MSB	Most Significant Bits
OCR	Optical Character Recognition
PCA	Principle Component Analysis
RBF	Radial Basis Function
RGB	Red Green Blue (color space)
RGB-D	RGB-Depth (Camera)
SVM	Support Vector Machine
YCbCr	Luminance-Chroma (color space)

ANR	Agence Nationale Recherche (France)
DGA	Direction Gnrale de lArmement (France)
IUT	Instituts universitaires de technologie
LISA	Laboratoire d'Ingénierie des Systèmes Automatisés
ISTIA	École d'ingénieurs de l'Université d'Angers
ESAIP	École Supérieure Angevine d'Informatique et de Productique
IMA	L'Institute de Management Accountants
LORIA	Laboratoire Lorrain de Recherche en Informatique et ses Applications
CAROTTE	CArtographie par RObоT d'un TErritoire

1

Introduction

Along with development of technology, automated systems and Robotics in particular, acquiring information from environment via sensory devices became a requisite in order to capacitate the systems of interacting with environment. These information can be interpreted from different data type, that's to say visual data (i.e. images, videos, depth medical scanning, etc.), voice signal, temperature signal, tactile, etc. Huge improvements have been reached for each data type, both on sensory device (hardware level) and interpretation methods. Simultaneously improvement of evolutionary methods inspired from biological systems, extended the study of this domain into a vast research area.

Capable of carrying lots of information and various options for implementation and interpretation, visual data drew the most attention. Mentioning some major fields of visual data applications such as Robotics, Medical applications, Human computer interactions, Automations and productions, ... will illustrate the importance of such a sensory data. As it's obvious, each mentioned field is composed of diversity of important and significant applications. Visual data itself can be treated with different techniques from different approaches like *Image Processing*, *Machine Vision* or *Computer Vision*, while each approach has different purpose, but yet they overlap in techniques and applications (Fig. 1.1). Image processing uses signal processing methods in general and more conventional methods, while computer vision uses more novel techniques. Signal processing is considered as a powerful tools for handling a lot of demands like *filtering*, *spectrum analysis*, *modulation*, *feature extraction*, *pattern recognition*, etc. Machine vision is an approach toward more specific demands in production and robotics field. It uses some techniques in order to obtain sufficient information for controlling an automated or robotic task.

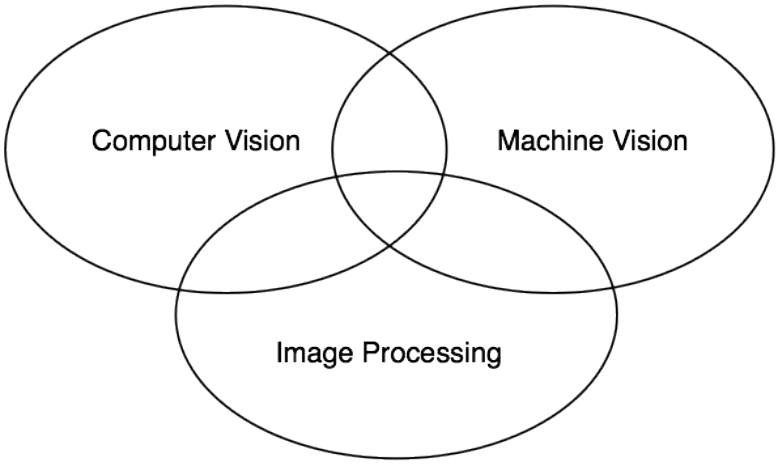


Figure 1.1: Handling visual data with different approaches.

Computer vision makes convenient tools available for variety of goals, by performing different tasks from *Image acquisition* and *feature extraction* to *segmentation* and *processing*. These tasks are executed toward different goals which may lie in fields of Automations, Control Processes, HCI-HRI, Medical Treatments and Robotics. Computer vision adapts the given task into one of the procedures of *Recognition*, *Motion Analysis*, *Scene Reconstruction* or *Image Restoration*. Motion analysis have different purposes like *optic flow* for robotics and control systems, *emotion analysis* for HCI-HRI or even *object tracking* for traffic analysis and etc. Image restoration does some fundamental functions like noise removal over images, and it's a mutual area with image processing. 3D modeling of the environment would be the result of scene reconstruction and useful for many applications. Recognition is considered as one of the most important demands, and it is handled by all computer vision, machine vision and image processing. Recognition may refers to different problems such as *Object Recognition*, *Pose Estimation*, *Pattern Recognition* and etc.

Since the purpose of this master's thesis was to develop methods integrating into CARTOMATIC project (section 1.2), some protocols and frameworks which already were stablished, should have been followed. An Important one affecting the goal of this work is a 3D model based comparison stage for object recognition. There exists a phase which will verify the output of classifier by comparing classified object with a predefined 3D model. Computational cost of this phase is high that each object could not be compared

with all 3D models, therefore a classifier should be realized to pick limited categories (ideally one category) and the result will be confirmed after 3D model based comparison.

This led to exploiting global features (Chapter 2) and realizing a multi-label classifier (Chapter 3). Objective of multi-label classifiers are usually to assign multiple labels per object were all of them are valid. For instance in music classification, a music track may fit in several classes, therefore all multi-labels would be valid [11]. But in the case of this project, multi-labeling schema is proposed to deal with ambiguity of the results, while global features are not sufficiently discriminative to reach an absolute decision based on them. In chapter 3 a classification method is proposed based on ANN and RBF networks with multi-labeling schema. Despite the problem of ambiguity, the original demand of this project would be a regular single-labeling classifier. Hence in addition, a single-label classifier called *Classomatic* have been developed and realized in chapter 4.

In first section of this chapter the problem of *Object Recognition* will be studied from different aspect of view, while literature will be reviewed for a better understanding of current state of art in this research area. In next section (1.2) the framework of this project which was a French Robotics National Competition called *CAROTTE* will be presented. Section 1.3 will review RGB-D cameras employed for 3D colored data acquisition.

1.1 Classification & Recognition

Recognition is a very important problem in all Computer Vision, Machine Vision and Image Processing areas. But not only for visual data, it is a vast domain dealing with variety of data types, including distinct problems such as *Visual Perception*, *Voice Recognition*, *Medical Image Analysis*, *Text Document Classification*, etc. For each type of problem numerous solutions have been proposed via different approaches, from conventional methods to evolutionary ones[8, 17].

Visual sensory data has a lot of usage in numerous problems such as medical treatments, Robotics , Human Computer Interaction, etc. In visual recognition area most cases are convolved with each other and they share mutual principles, hence the solutions are fundamentally similar. If one wants to list some important areas of visual recognition with respect to literature,

this is a possibility:

- Pattern classification: it is a more general case which dominates others like face recognition and handwriting recognition, nonetheless it is useful for disease diagnosis, text document classification, speech recognition, etc.[8].
- Handwriting recognition (like OCR): it is mostly useful for digitalizing text documents, in order to facilitate searching and electronic publications. It's also used in HCI.
- 3D object recognition: it is a vast area and very useful in robotics, HCI, etc. It is the procedure of detection and identification of an object in environment.
- Face recognition: it is applied in many areas, like photography, document analysis, criminology, security, HCI and etc.
- Pose estimation: it means to identify the position and orientation of an object with respect to a reference frame. It is useful in robotics for manipulating objects, localization (when the object is a land mark), obstacle avoidance, HCI (when the object is a human body) etc.
- Motion analysis: unlike regular cases here a stream (sequence of images) is processed, useful in HCI, HRI, etc.
- ...

1.1.1 Object Recognition Approaches

Object Recognition is a vast area with lots of applications and lots of approaches. Some applications are already mentioned, and here different approaches will be reviewed.

1. *Geometric model* of objects was the first clue to distinguish their appearance variation due to viewpoint change. The main idea is that the geometric description of a 3D object allows the projected shape to be accurately predicated in a 2D image under projective projection, thereby facilitating recognition process using edge or boundary information.

2. *Appearance-based* techniques as advanced feature descriptors were developed. Among all *eigenface* could be considered as the most outstanding method. The underlying idea of this approach is to compute eigenvectors from a set of vectors where each one represents one face image as a raster scan vector of gray-scale pixel values. Each eigenvector, dubbed as an eigenface, captures certain variance among all the vectors, and a small set of eigenvectors captures almost all the appearance variation of face images in the training set. The eigenface approach has been adopted in recognizing generic objects. As the goal of object recognition is to tell one object from the others, discriminative classifiers have been used to exploit the class specific information. Classifiers such as *k-nearest neighbor*, *neural networks* with *radial basis function* (RBF), *dynamic link architecture*, *Fisher linear discriminant*, *support vector machines* (SVM), *sparse network of Winnows* (SNoW), and *boosting algorithms* have been applied to recognize 3D objects from 2D images[9].

3. *Feature-based* methods should be mentioned, in which *interest points* are defined and play the main role. Interest points are those at intensity discontinuity, that are invariant to change due to scale, illumination and affine transformation. The *scale-invariant feature transform* (SIFT) descriptor proposed by Lowe[16], is the most significant method used for feature representation. "Although the SIFT approach is able to extract features that are insensitive to certain scale and illumination change, vision applications with large base line change entail the need of affine invariant point and region operators"[9].

It is also possible to classify these methods other ways. For instance *Peter M. Roth* and *Martin Winter* surveyed methods in a technical report[6] and split them into two categories, namely *local* and *global* approaches. Local approaches search for salient regions characterized by e.g. corners, edges, or entropy. They regard characteristics of objects such as the color, gradient, etc. Local approaches itself divides into *Region of Interest Descriptors* and *Region of Interest Detectors*. In contrast to that, global approaches model the information of a whole image like histogram. In a later stage, these regions are characterized by a proper descriptor. For object recognition purposes, obtained local representations of test images are compared to the representations of previously learned training images[6].

1. Local approach
   Most important methods of *interest detector* could be listed as *Har-*

ris or Hessian point based detectors (Harris, Harris-Laplace, Hessian-Laplace), *difference of Gaussian points (DoG) detector*, *Harris- or Hessian affine invariant region detectors* (Harris-Affine), *maximally stable extremal regions* (MSER), *entropy based salient region detector* (EBSR) and *intensity based regions and edge based regions*. All *interest detectors* fall into three categories based on their methodologies and attributes:

- corner based detectors
- region based detectors
- other approaches

As well most important methods of *interest descriptors* are listed as *SIFT*, *PCA-SIFT* (gradient PCA), *gradient location-orientation histograms* (GLOH),*Spin Images*, *shape context*, *Local Binary Patterns*, *differential invariants*, *complex and steerable filters*, and *moment invariants*. The *interest descriptors* are classified as:

- distribution based descriptors
- filter based descriptors
- other methods

2. Global approach
Discussion over the global approach will be reduced to subspace methods. These methods generally perform a mapping from input image into a lower dimensional subspace more suitable for each different task. Some important methods could be itemized as:

   (a) *Principal Component Analysis*: "The main idea is to reduce the dimensionality of data while retaining as much information as possible. This is assured by a projection that maximizes the variance but minimizes the mean squared reconstruction error at the same time"[6]. With respect to each tasks and based on the critics it can be performed differently like *Derivation of PCA*, *Batch Computation of PCA*, *Efficient Batch PCA*, *PCA by Singular Value Decomposition* , etc.

   (b) *Non-negative Matrix Factorization (NMF)*

   (c) *Independent Component Analysis*

   (d) *Canonical Correlation Analysis*

   (e) *Linear Discriminant Analysis (LDA)*

   (f) *Extensions of Subspace Methods*

1.1.2 Challenges in object recognition

There are number of problems appearing in object recognition, as well there exist plenty of literatures studying them and proposing solutions. Most important and discussed topics will be reviewed here:

1. *Overlapped and Occluded Objects*:
   "In traditional model-based object recognition systems, a model of each object in the model database is matched in a random sequence against the scene image. The matching procedure must be repeated for every model in the database until a correct match is found."[39] The problem which is not specific to traditional methods is occluded and overlapped objects, in real world there is no guarantee for objects in acquired image to be straightforward and in plain site. Some works done for this problem will be reviewed. *Peter W. M. Tsang* and *P. C. Yuen* developed and reported a computer vision system for recognition of real world images [44]. The approach, basically, removes the dependence of an object representation to its spatial parameters with the use of difference chain code (DCC) in contour encoding. An object shape is identified by the system through the detection of selected discrete feature segments in the contour code instead of attempting to search for a complete boundary. They claimed the system is capable of identifying multiple overlapped objects in a scene without stringent restrictions on their size, shape and orientation. *Jung H. Kim*, *Sung H. Yoon* and *Kwang H. Sohn* proposed a method to decrease the computational effort and increase robustness of the system [36]. A Hybrid Hopfield Neural Network (HHN) algorithm which combines the advantages of both a Continuous Hopfield Network (CHN) and a Discrete Hopfield Network (DHN), was described and applied for partially occluded object recognition in a multi-context scenery. *Du-Ming Tsai* and *Ray-Yuan Tsai* presented an Artificial Neural Network (ANN) approach to determine the matching order of models in the database [39]. The match between a given scene image and a model is based on the rank of similarity of the model rather than its serial storage order in the database. Both isolated and overlapping objects that comprise piecewise linear and circular segments are considered for the recognition. *Jiann-shu Lee* et al. proposed a method to recognize partially visible two-dimensional objects by means of multi-scale features based on Hopfield neural network [34]. The Hopfield network was employed to perform global feature matching. *P. N. Suganthan*, *Eam Khwang Teoh* and *Dinesh P. Mital* proposed an energy formulation for homomorphic graph matching by the Hopfield network and a Lyapunov indirect

method-based learning approach [27]. The adaptation scheme eliminates the need to specify the constraint parameter empirically and generates valid and better quality mappings than the analog Hopfield network with a fixed constraint parameter. *Ji-Xiang Dua* et al. proposed a method for recognition of 2D occluded shapes based on neural networks using generalized differential evolution training algorithm[10].

2. *Orientation of objects in aspect of view*:
Considering real world, it's clear capturing images all from same angle as the images stored in databased or the images used for system training is not feasible, therefore the angle view of the object in the image is going to vary. Although it seems not to be a problem for symmetric object, still the light and shadow over the symmetric object will effect the appearance of it, also symmetric object are very rare case. Some works have been done and the orientation of the object have been taken care of mostly as a side issue.[22, 3]

3. *Resolution, Size and noise of input data*
Generally systems are supposed to be robust and tolerate noises, sometimes the size and resolution of the images in visual data acquisition may cause problem. "Noise and sparsity of data in the imaging context result in degradation of quality of the reconstructed image as a whole, instead of affecting it in the form of local corruption of the image pixel information as in many image processing situations"[42], Hence:

   - neighborhood processing methods for noise removal may not be suitable.
   - feature extraction cannot be reliably performed.
   - model-based methods for classification cannot easily be applied.

   A. Ravichandran and *B. Yegnanarayana*[42] studied the performance of artificial neural network models for recognition of objects from poorly resolved, noisy, and transformed (scaled, rotated, translated) images, such as images reconstructed from sparse and noisy data in a sensor array imaging context. They demonstrated that neural network models can be used to overcome some of the difficulties in dealing with degraded images as obtained in an imaging context.

1.2 CAROTTE the Competition

Whether in nature or urban, it is common for the environment in which robots can not be calibrated easily, this uncertainty may jeopardize the achievement of missions, including those concerning the exploration of hazardous areas.

In this context, small unmanned ground vehicles (UGVs) can be used to help humans based on their recognition capabilities. One of the key abilities of these robotics systems is their ability to gather and analyze information about their environment to provide information on the configuration of the place (mapping) and recognition and localization of interesting objects. The maximum autonomy of the robots must be achieved along with the robustness of the system, for example like the case of communication disruptions.

To improve *localization abilities*, *mapping of buildings* and *terrain analysis* in urban areas, the DGA and ANR have launched a challenge called CARROTE[1].

Objectives of this challenge were:

- To advance innovation and state of the art in robotics' perception/cognition for dual applications (defense and security, emergency preparedness, home care, companion robot).
- Excite connections between researchers and industrialists from robotics and related areas (augmented reality, game, image analysis, indexing, semantic ,...).

This challenge tested the capacity of small ground robots for reconnaissance missions in a closed environment not totally structured. Innovations were expected in the field of artificial intelligence embedded (perception, recognition, data fusion, semantic mapping, indoor localization, control and autonomy architecture) with the potential for advances in other areas (mobility, planning Mission and supervision, human-computer interfaces, ...).

The Competition: Each team carried out an autonomous robotics system, able to move in a confined space and recognize objects in this room,

[1] http://www.defi-carotte.fr/

to provide a mapping accompanied by semantic annotations of the unknown space. The challenge was realized in three phases over three years:

1. *First phase (9 months)*: realization of these systems. It was ended with the first event in mid 2010, followed by an analysis of results.

2. *Second phase (1 year)*: further developments in order to participate in a second competition (mid 2011) more complex (involving hazards, less structured environment).

3. *Third phase (1 year)*: ultimate system improvements for the final competition (June 2012) including more complex situations to be analyzed (new types of hazards ...).

Challenges in this competition was *Autonomous Navigation* (450 points), *2D Map Creation* (400 points), *3D Scene Construction* (400 Points), *Wall-Floor Classification* (500 points), *Object Recognition* (2500 points).

LISA was the host team at the University of Angers, founded in 1990. After a long period during which LISA has focused on the theme of *Discrete Event Dynamical Systems*, now there is also a team working on *Signal and Image*. With conjoining new colleagues, LISA now is active in other themes related such as *Hybrid Dynamical Systems*, *Virtual Reality* and *Robotics*. The LISA is divided into two teams interconnected:

- Modèles et Systèmes Dynamiques (MSD)

- Signal-Image (SI)

So far, LISA has become a laboratory of the University of Angers in it's own right that is to brings together colleagues from five components of university of Angers (*ISTIA, IUT, UFR Sciences, IMIS-ESTHUA, Médicine*) as well as the *ESAIP* and *IMA*. The laboratory is involved in the *Pôle Math-STIC*.

A consortium called *CARTOMATIC* was made up by LISA and *LORIA* participating in CAROTTE competition.

3D object recognition : is defined as a part of CARTOMATIC project in the framework of CAROTTE competition. Third phase of this project include the ultimate development of environment consisting object recognition, and LISA's team is responsible for this task in the CARTOMATIC consortium. This phase would be the concern of this master's thesis. Objects to recognized during the competition are categorized as follow:

- Chairs.
- Paper Boxs.
- Books.
- Glasses/Mugs.
- Electrical Elements.
- Bottles.
- Containers.
- Tools.
- Boots.
- Canonical Objects.
- Keys.
- Cameras.
- Fans.
- Radio.
- Toys.
- Plant.
- Fruits.
- Robot.
- Computer Keyboards.
- Drawer.

Some objects appearing in the competition were announced before and some other new objects falling in the same categories were presented during the competition.

1.3 RGB-Depth Cameras

Novel sensors called *RGB-D cameras* are coming to attention more and more everyday, because of rich information they provide from a scene with a single shot. They provide "RGB" images along with a *depth* image which letter "D" stands for. This combination makes the acquisition potentially a 3D colored image as illustrated in Fig. 1.2c and Fig. 1.3. A popular solution for realization of such a device is the *KINECT*™.

Fig. 1.2 demonstrates raw output of the RGB-D camera and illustration of 3D image forged by combination of depth and color information. Calibrating the kinect with a fixed frame on the robot, each point in depth map will turn to a 3D Cartesian coordinate. A 3D colored image will be constructed by assigning those coordinates to corresponding pixels in RGB image. Thus one point of the 3D cloud is composed of six parameters:

1. X-coordination
2. Y-coordination
3. Z-coordination
4. RED channel
5. GREEN channel
6. BLUE channel

(a) Depth image (b) RGB image (c) 3D image

Figure 1.2: Depth and RGB images are merged into colored *Point Cloud*.

Fig. 1.3 presents another sample of 3D colored scene illustration.

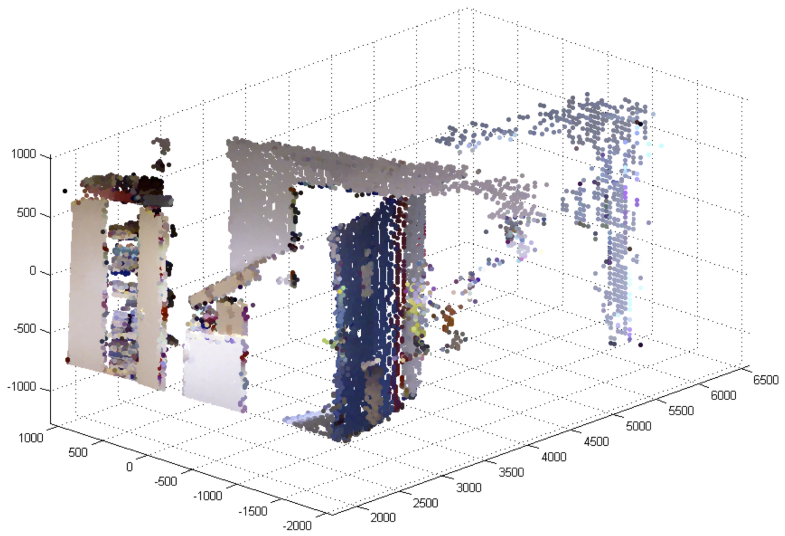
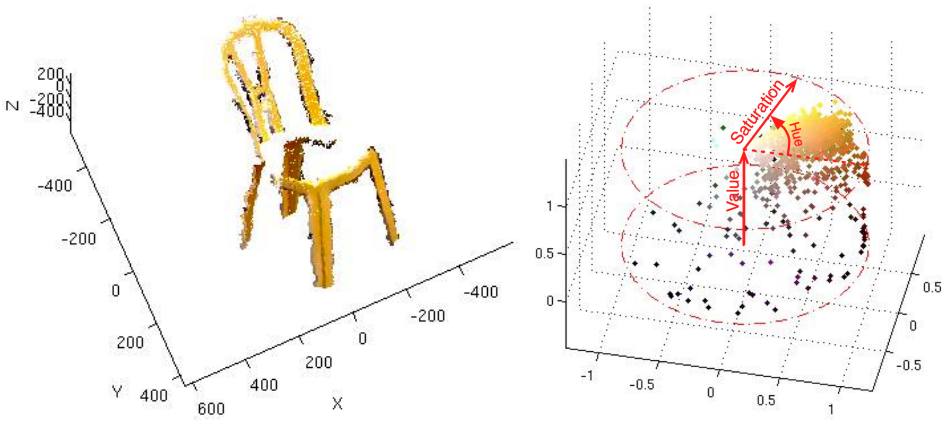


Figure 1.3: 3D model illustrated from 3D acquired data.

2

Features Selection

Two general sets of information are at hand for every single point of an object, *position* and *color* each in 3 dimensional space as demonstrated in Fig. 2.1. The color space is chosen to be *HSV* for demonstration (and further use).



(a) Illustration of the 3D point cloud of a segmented chair

(b) points distribution in HSV colorspace

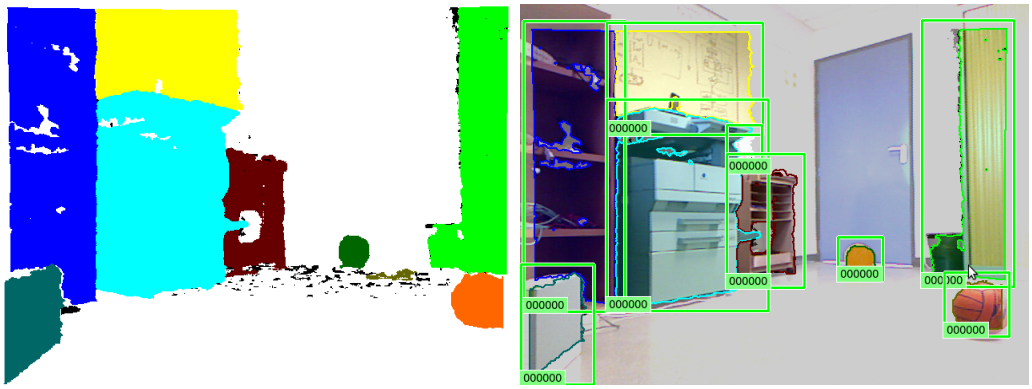
Figure 2.1: Illustration of two set of information obtained via Kinect.

In first section the idea of object detection will be reviewed. Since each scene may contain several objects, It will be explained how all *potential* objects in a scene are detected and isolated. Following section will detail the feature selection, where Global Features will be surveyed. Exploiting global features, rises the need of evaluating different color spaces to find the most discriminant one. *Principle Component Analysis* will be mentioned as a very

useful tool for features projection during this chapter, while computational notes are given in Appendix A.

2.1 Object Segmentation

Acquired data from the scene contains depth information auxiliary to color information which makes the isolation of the objects easy. First step would be *Floor Removal* based on the cartesian coordinations of the points obtained through calibration mentioned in section 1.3. Those points below a threshold (2.5^{cm}) in Z-axis are considered as floor and will be removed from the scene. After that dependency of segmentation algorithm is on the depth information, those points connected to each other without a gap (Fig. 2.2a) make a region which is potentially an object (Fig. 2.2b). During learning process value of the *gap* may varies from one object to another based on their size, while this gap value will be compromised to a fix value for the recognition phase where segmentation shall be done automatically.



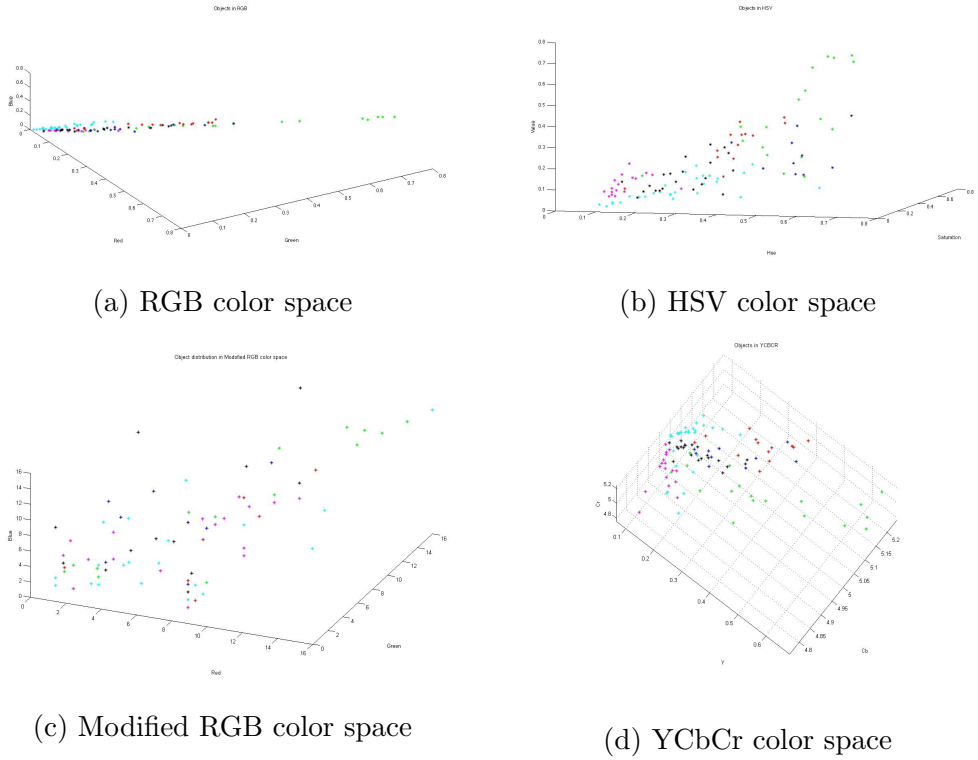
(a) Segmented regions (b) Isolated *potential* objects

Figure 2.2: An example of object *detection* process.

A sample of scene segmentation (i.e. *object detection*) is illustrated in Fig. 2.2. All isolated regions will be considered potentially as objects until they are assessed in the classification phase.

2.2 Color Spaces

The color space is chosen to be *HSV*, that's because after several attempts on different color spaces such as *RGB*, *YCbCr*, and even their modifications, none were as distinguishing as *HSV* which is known to be more light-independent. Not only the distribution of objects from different classes regarding their average colors gives the idea that none of the color spaces are as distinguishing as HSV, but also it has been evaluated in practice by some experiments. Fig. 2.3 realizes the intuition of HSV being prior to other color spaces.



(a) RGB color space

(b) HSV color space

(c) Modified RGB color space

(d) YCbCr color space

Figure 2.3: Distribution of 107 objects from 6 different classes in 4 color spaces.

Due to light-dependency of RGB color space, the average value for an object may not be a good feature to rely on. Instead in each color channel the histogram will be plotted and the peak of each will be chosen for representing that channel. To avoid the effect of exposure problem mentioned, 4 MSBs of each color channel will be considered. This operation is equal to

dividing the values of each color channel by 16.

$$
\begin{aligned}
R = \underbrace{0010}_{\text{MBS}}\,0110 \quad &\longrightarrow \quad R = 0010 \\
G = \underbrace{1001}_{\text{MBS}}\,1100 \quad &\longrightarrow \quad G = 1001 \\
B = \underbrace{0011}_{\text{MBS}}\,0001 \quad &\longrightarrow \quad B = 0011 \\
0 \leq RGB \leq 255 \quad &\longrightarrow \quad 0 \leq RGB \leq 15
\end{aligned}
$$

To calculate the modified histogram, values of RGB channels will be divided by 16 and the number of pixels with same value will present a column on new histogram. This histogram will hand a modified version of RGB color space presented in Fig. 2.3c. In modified RGB color space instead of average color the peak column in histogram will be picked (Fig. 2.4).

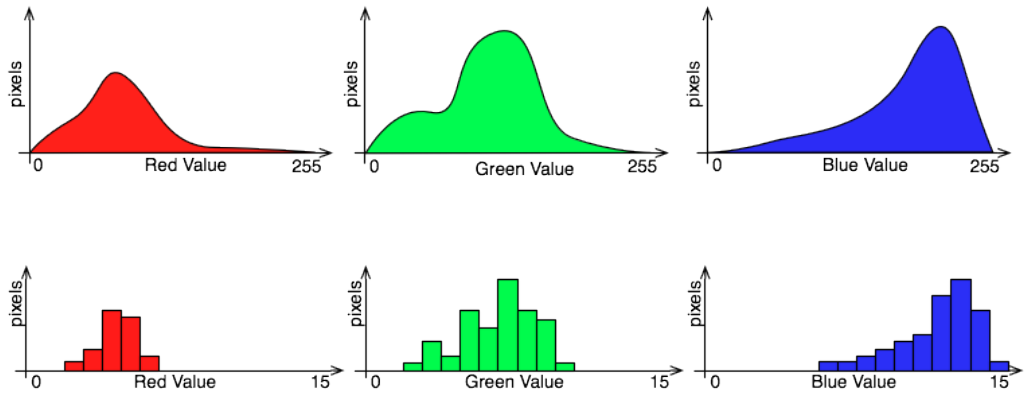


Figure 2.4: Modification of RGB color space, by modification of the histograms and collecting the peak instead of average value.

After all, none of color spaces behave as discriminant as HSV, even modified version of RGB. Comparison of HSV and RGB is illustrated in Fig. 2.5.

Note that during all the calculations *cylindrical coordinate system* of HSV colorspace must be taken into account. To facilitate the calculations, coordinates of colors have been converted from cylindrical to cartesian coordinate system in the same HSV color space. This will help avoiding problems coming from angular value of *Hue* parameter, like those points close to 0 and 2π which are actually close to each other, but far away regarding their angular values. This conversion will ease the use of Mahalanobis or Euclidian dis-

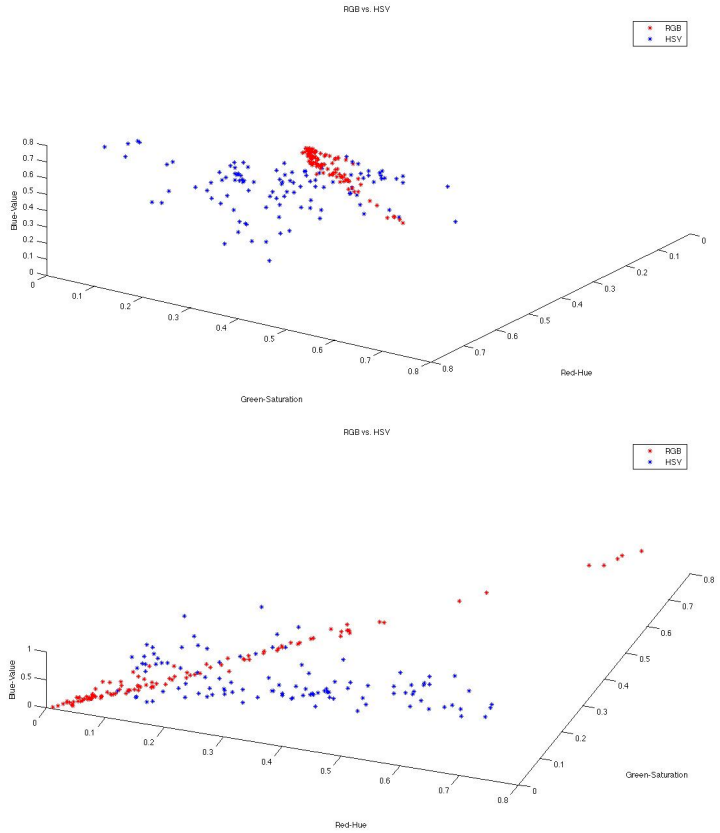


Figure 2.5: Comparing RGB and HSV distribution of 107 objects from 6 classes in . RGB in red and HSV in blue. As visualized, objects in HSV are more differentiated since they are distributed more dispersive.

tance, as well as variance calculations.

2.3 Feature Extraction

As mentioned previously (section 1.3) the goal of this project is to pick closest categories for each unknown object and there will be a 3D model based verification phase afterward. Since the shape based verification some how covers the local features, it's reasonable to construct classifiers based on global features. As well the intrinsics of the RGB-D cameras bring the global features of an object to attention, hence only global features have been studied and exploited during this work.

Based on those two characteristic spaces obtained via kinect acquisition, four global feature sets are proposed to be extracted:

- size of smallest bounding box containing the object (3D),
- variance of points in position space (3D),
- average color of all the points in object (3D),
- variance of color of all points (3D).

PCA[1] (*Principal Component Analysis*) is well known to be a popular feature extracting method [24] both in conventional recognition algorithms and Neural Network based methods. For each isolated object PCA calculation will be performed in position space and the object will be rotated in a way that 1st principle component aligns with X-axis, 2nd with Y-axis and 3rd with Z-axis. With this orientation of object, the problem of finding smallest bounding box will decrease to minimum and maximum values of all the points along 3 main axis (X, Y and Z). PCA calculation of position of the points provides variance of mass presenting a significant characteristic of the object. Variance of points in position space will be simply those standard deviations already obtained via PCA calculation.

After that a translation from RGB color space to HSV have been performed, the feature extraction from color space will begin. It should be noted, in order to facilitated the color space calculations, after the translation to HSV the coordinates of points in color space will be defined by cartesian not cylindrical coordinate system. To acquire features from color space, first an average of all points will be calculated, and then the variance of points in color space along 3 axis (cartesian coordinate system) will be acquired.

This procedure will provide 4 vectors of features, each in 3 dimensions:

$$Feature\ Vector = \left\{ \begin{array}{l} Size\ of\ Bounding\ Box\ (3D) \\ Variance\ in\ Position\ Space\ or\ Mass\ (3D) \\ Average\ Color\ (3D) \\ Variance\ in\ Color\ Space\ (3D) \end{array} \right\}$$

It worth to mention that these 4 vectors are independent from each other. Depending on the structure of classifier these vectors could be engaged all together as a 12 dimension vector, or could be employed separately.

[1] Calculations of PCA is detailed in Appendix A.

2.4 Splitting Object for Extra Features

Although the reason for choosing global features was that the objective is to pick even several label per object (multi-labeling), yet it's reasonable to set up a technique which produce auxiliary feature vectors for disambiguation and absolute classification. This comes available by splitting the object in smaller pieces (Fig. 2.6) and calculating the same feature vectors mentioned in section 2.3 for each piece.

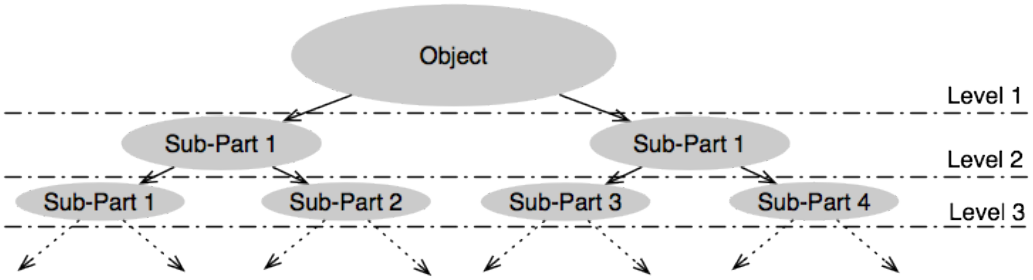


Figure 2.6: Splitting an object into small pieces

Different levels obtained in Fig. 2.6 could be assigned with different weights for their roles in classifications. A simple weighting strategy is presented in equation 2.1, where w_{level} represent weights for each level.

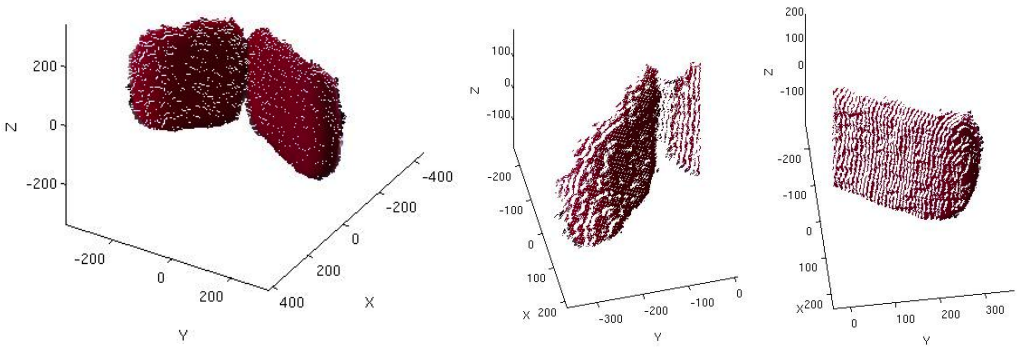
$$w_{level} = 2^{-level} \tag{2.1}$$

There exist multiple sub-parts at each level, which means the weight of each level should be shared between them. When number of sub-parts depends on the level ($2^{level-1}$), weight for a sub-part at a level is given in equation 2.2.

$$w_{sub-part} = \frac{2^{-level}}{2^{level-1}} = 2^{(1-2\times level)} \tag{2.2}$$

Splitting could be done regarding different conditions, but here only the mass of object is taken into account. After moving the object to origin of base frame, principle components of the object will be calculated and will align to axes of the base frame (X,Y,Z). Those points having x value more than zero ($x > 0$) will construct first sub-part and the rest will construct the second sub-part. Fig. 2.7 demonstrates one level of splitting of a *chair*

which only seat and back was obtained during acquisition. Fig. 2.7a shows the object in *level 1* (before split), and two sub-parts of *level 2* are given in Fig. 2.7b and Fig. 2.7c.



(a) level1 (b) level2 - sub1 (c) level2 - sub2

Figure 2.7: Splitting a *chair* with respect to 1st *principle component*.

3

Multi-label Classification with ANN

Not only artificial neural networks but the real neuron systems of the human brain is still an open field of research. Artificial neural networks tend to simulate the information processing ability of biological neural networks, although they are only one of the possibilities in information processing. Early model of artificial *neuron* was proposed by Warren McCullach and Walter Pitts at 1943. "The main difference between neural networks and conventional computers is the massive parallelism and redundancy"[35], and the adjustment of parameters occur through training not programming. Artificial neural networks vary in structure of the nodes, topology of the network and learning methods with respect to their applications[24, 17, 4, 37, 33, 15].

In order to realize the information processing capability of Artificial neural networks *Raùl Rojas* proposed Fig. 3.1, presenting ANN against other computational models [35]. For a better understanding of Artificial Neural Networks, a brief description of each model will come next.

Mathematical Model: Function's computability came to discussion since the beginning of this century, after a satisfactory definition of *computability* was given. The idea starts with some primitive functions and composition rules that are computable, the rest of the functions obtained from primitive functions and composition rules are then computable [35].

Logic-Operational Model (Turing Machines): Alan Turing introduced another kind of computing model with advantage of consisting an opera-

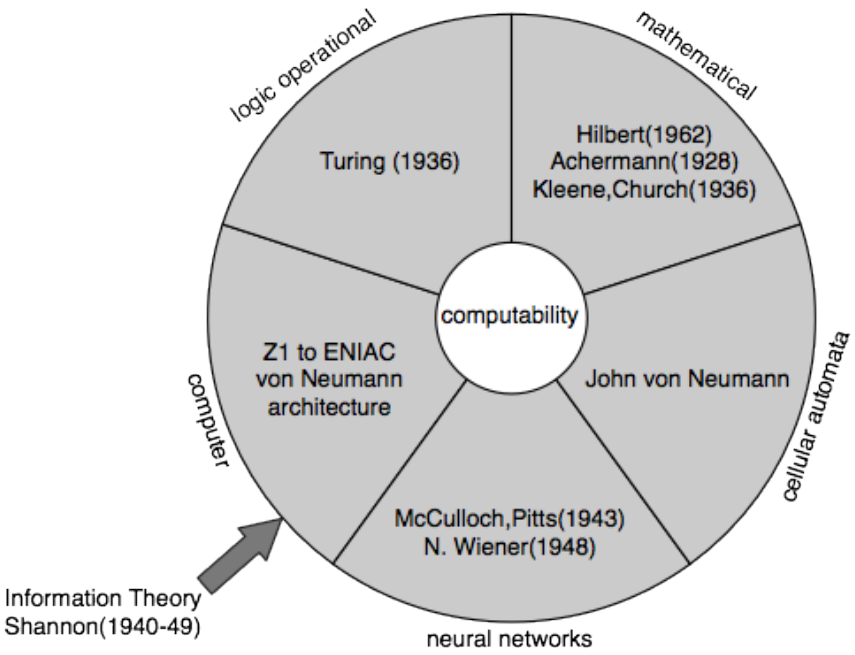


Figure 3.1: Five model of computation [35].

tional, mechanical model of computability. For the first time by his proposal the meaning of *programming* became clear [35].

Computer Model: Early In the 1930s and 40's engineers were not aware of Turings research completely, yet they developed the first electronic computers which the term *computation-with-the-computer* could mean *computability*. Primitive computer first became able to process a sequence of instructions, eventually they were developed to a state which could perform iterations. Even execution of iteration was not open-ended at first ("while" loop) but only possible for a specified times ("for" loop). Later *Mark I* was developed as the first universal computer able to cover all computable functions by making use of conditional branching and self-modifying programs, which is one possible way of implementing indexed addressing [35].

Cellular Automata: This model resembles kind of a massively parallel multi-processor system. To define a minimal architecture of a universal computer *John von Neumann* analyzed a new computational model which he called *Cellular Automata* and could operate in a "computing space" in which all data can be processed simultaneously. All computable functions, in the sense of Turing, can also be computed with cellular automata [35].

Biological Model: Neural networks do not operate sequentially, as Turing machines do. Neural networks have a hierarchical multilayered structure which sets them apart from cellular automata, so that information is transmitted not only to the immediate neighbors but also to more distant units [35].

3.1 Modeling Neural Networks

Most important issues in modeling the neural networks and constructing artificial models of them is the definition of the *primitive functions* and *composition rules* of the computational model. These are primary task of simulating the natural neural system (Fig. 3.2). Regarding conventional Von Neumann model of computing for the artificial neural networks, the primitive functions are located in the nodes of the network and the composition rules are contained implicitly in the interconnection pattern of the nodes.

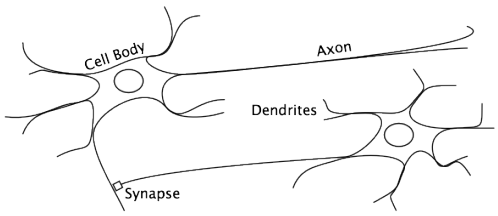


Figure 3.2: Natural Neurons; Cell body is modeled with *primitive functions* while Axon and Synapse are modeled by *composition rules*.

Primitive function of neural networks is presented as a function with several real-value signals as input and one output, where the function itself is selected arbitrarily, as presented in Fig. 3.3. Usually each input signal x_i is associated with a coefficient called weight w_i.

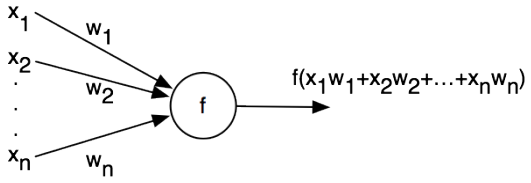


Figure 3.3: *Primitive Fucntion* as neuron model.

McCulloch-Pitts proposed the model of neuron (primitive function) as accumulation of weighted inputs followed by a threshold function (Fig. 3.4).

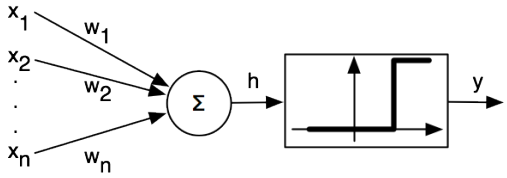


Figure 3.4: *McCulloch-Pitts* model of neuron.

One of the most typical structure of the network is presented in Fig. 3.5. By such a structure the network is considered as a function Φ evaluated at the point (x_1, x_2, x_3). Φ is called network function and is performed by convolving four nodes each executing a primitive function (f_1, f_2, f_3, f_4). Functionality of network function Φ presented in Fig. 3.5 is expanded in equation. 3.1.

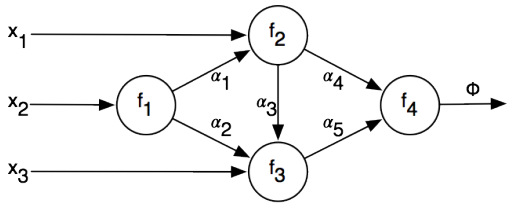


Figure 3.5: A typical functional model of artificial neural networks[35].

$$\phi = f_4(\alpha_4 f_2(x_1, \alpha_1 f_1(x_2)), \alpha_5 f_3(x_3, \alpha_3 f_2(x_1, \alpha_1 f_1(x_2)), \alpha_2 f_1(x_2))) \quad (3.1)$$

Until here two main issues of the artificial neural networks are covered; definition of primitive functions and interconnections between them (topology of the network). With the same network topology and primitive functions, network function Φ could perform differently with different weights.

Here is where *learning* process take action to adjust the network for each specific demand. It is worth pointing out in order to deal with real world problems successfully, not only the primitive function, network structure and learning methods are important, but there are other issues which should be taken care of such as network size and number of training samples.

3.2 RBF Networks

Radial Basis Function networks (Fig. 3.6) have been employed for several purposes from function estimation to control. These networks have been employed to construct a multi-label classifier[1]. As demonstrated in Fig. 3.6, RBF networks are composed of 3 layers of neurons; *input* layer, *hidden* (RBF) layer and *output* layer.

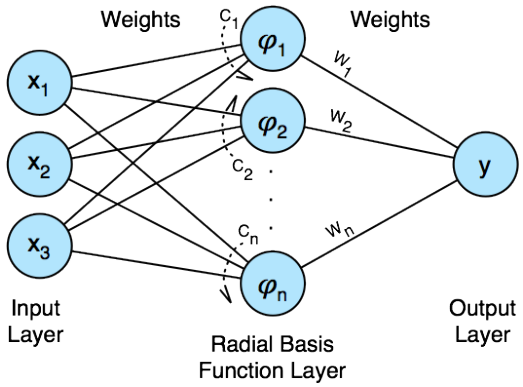


Figure 3.6: Single RBF Network.

Those weights from input layer to hidden layer are not multiplicative but subtractive representing the centers of RBF functions, while output weights are multiplicative in role of amplitudes of RBF functions. Mathematic definition of the network illustrated in Fig. 3.6 is expressed by equation 3.2.

$$y = RBFN(X) = \sum_{j=1}^{n} (w_j \times \varphi(X - C_j)) \tag{3.2}$$

In equation 3.2, y is the output of the network, X represent input vector (containing x_1,x_2,...), $RBFN$ is the functionality of network resulting y, w_j stands for those weights from hidden layer to output node, φ represents the

RBF functions in hidden layer and finally C_j represent the center of RBF functions. φ functions only differ in the position of center and amplitude.

3.3 Classification Methodology with RBF Networks

Purpose of this work is to assign closest label (or labels) of different categories to an unknown object. This will be realized by estimating distributions of different classes in each feature space (e.g. size, color, etc.). Comparison of unknown object's features with these estimations would give an idea if the object belongs to any class (or classes) or not. Distinctive RBF networks have been employed for estimating each feature space, where *Gaussian* functions are recruited in hidden layers as RBF functions and that is expressed in equation 3.3.

$$\varphi_j(X - C_j) = \exp\left(-\frac{\sum_{k=1}^{K}(x_k - c_{jk})^2}{2\sigma^2}\right) \tag{3.3}$$

In equation 3.3, $\varphi_j(X - C_j)$ is the output of Gaussian function in hidden node j and C_j are those weights from input layer to that hidden node representing the center of Gaussian function, K is the dimension of input space, and σ is the standard deviation of the Gaussian function. The output of each network which represents the probability of a point being included in that specific distribution, will be followed by thresholding function at 0.5 to provide a binary output (as shown in Fig. 3.7).

Membership of an object in different classes will be evaluated by individual classifiers unbiased by other classes (Fig. 3.7). Proposed method for classification here is a multi-label assigning scheme in which every input will be assigned with either 0, 1 or even more labels at the same time (Fig. 3.8), while only one of them could be correct. With such a scheme two type of error may occur:

- *False Positive*: where the object is assigned with correct label plus extra labels (*Multi-Labeling*).
- *False Negative*: when an object is *not* assigned with correct label.

Although there should be a compromise between these two types of error, but it should be mentioned that the *False Positive* is remediable with further

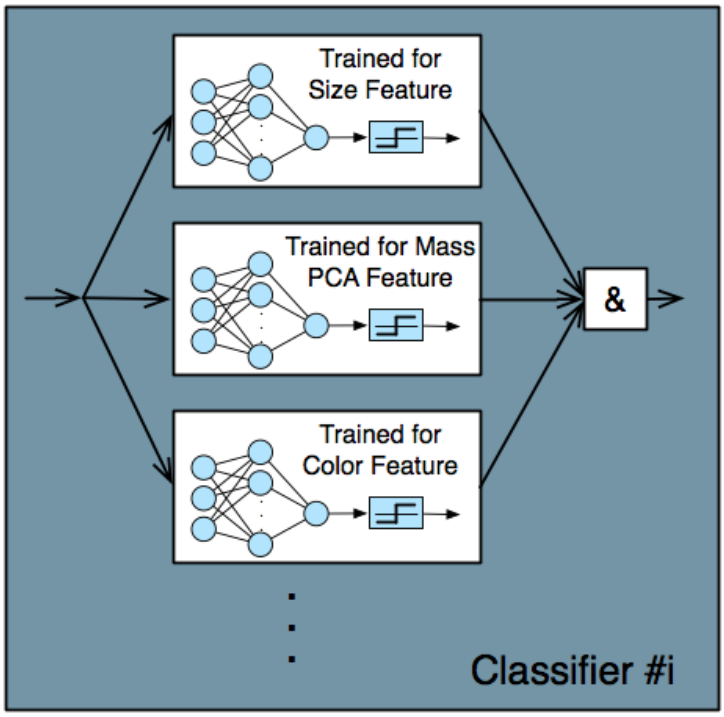


Figure 3.7: A classifier trained with objects from *Class#i*.

phase after classification relying on local features of the object (shape based verification).

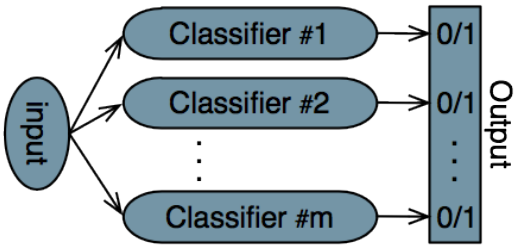


Figure 3.8: Multiple label assignment schema.

To verify membership of an object in a specific class, the object will be examined in all different feature spaces (e.g. size, color, etc.) separately by distinctive RBF networks in each feature space. Each network is trained in one feature space and membership will be confirmed if all networks in different feature spaces approve. Therefore totally "$m \times n$" distinctive networks will be constructed, where m is the number of classes and n is the number

of feature spaces. Key idea of this classification is expressed in Algorithm 1.

Algorithm 1 RBF Networks Classification Methodology

```
for all Classes do
  membership = 1;
  for all Feature Sets do
    Introduce the object to the specific RBF network trained for current
    "class" and current "feature set".
    if RBF(object) < 0.5 then
      membership = 0;
    end if
  end for
  if membership == 1 then
    Assign the label of current class to object.
  end if
end for
```

Although proposed algorithm in current layout picks multiple labels for each object, makes it more disposed as a pre-classifier demanding a further stage of accurate decision, it could be adjusted to operate stand alone if the feature search space is discriminant enough. Such an adjustment takes place by ignoring the "*thresholding by 0.5*" operation mentioned earlier in this same section, where analog outputs of the RBF networks stand for the probabilities of an object being a member of each class. Available probabilities makes it possible to pick the highest as winner.

One of the important characteristics of this network (along with training algorithms presented in sections 3.4 & 3.5) is it's independency of *negative reinforcement*. This characteristic will help to set up independent classifiers for different classes as it was explained, unlike conventional classifier methods for multi-class problems where they are dependent to other classes like *one versus all* or *one versus one*.

3.4 Conventional Learning Algorithm

There exist a classical algorithm for training RBF networks proposed by *C. M. Bishop* which is only *input-driven* [41]. *G. Bugmann* presented a modified training algorithm (Algorithm 2) which is both *input-driven* and

output-driven [29]. With help of modified training algorithm the number of nodes in hidden layer decreases (i.e. less resources) without compromising the performance. The process of introducing training points to the network is iterative in Algorithm 2.

Algorithm 2 Modified Training Algorithm
for a Gaussian RBF Network

```
for "Epoch" times do
  for all point_i ∈ Training Dataset do
    error = output^desired − GRBFN (point_i)
    GRBFN is the network under current training process.
    if error < tolerance then
      Do Nothing.
    else
      Find the closest Gaussian function(φ) in the hidden layer and compute it's distance to point_i.
      if distance < threshold then
        Update that closest node.
      else
        Recruit a new node in hidden layer centered on point_i.
      end if
    end if
  end for
end for
```

Recruiting: to generate a new node in hidden layer, where the input weights (center of RBF) are initialized by the introduced input itself, and the output weights will be initialized by 1, to be updated in later epochs.

Updating: as input weights represent centers of Gaussian functions, they will be updated toward the introduced input (equation 3.4), while output weights behave as amplitudes of Gaussian functions and they will be updated considering the error occurred in output (equation 3.5).

$$c_j(t+1) = 0.8c_j(t) + 0.2x \tag{3.4}$$

$$w_{ij}(t+1) = w_{ij}(t) + learnrate \times (y_i^{desired} - y_i^{real}) \tag{3.5}$$

As *G. Bugmann* proposed, to ensure covering the area between training points, the variance of Gaussian function (σ) should be close to the "half of the largest distance between nearest neighbors" as in equation 3.6, where N is the number of objects [29].

$$\sigma \rightarrow \frac{1}{2} MAX_{i=1}^{N} \left(min_{\substack{j=1 \\ j \neq i}}^{N} \left(distance(object_i, object_j) \right) \right) \tag{3.6}$$

Illustrated in Fig .3.10 is the performance of this Gaussian RBF networks in 2D distribution (function) estimation, trained with presented learning algorithm. Estimated 2D distribution is based on a plateau function shown in Fig .3.9.

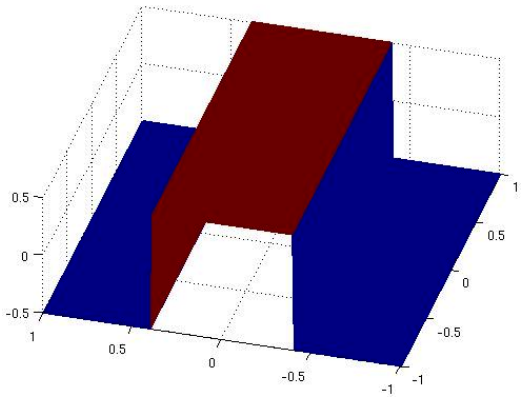
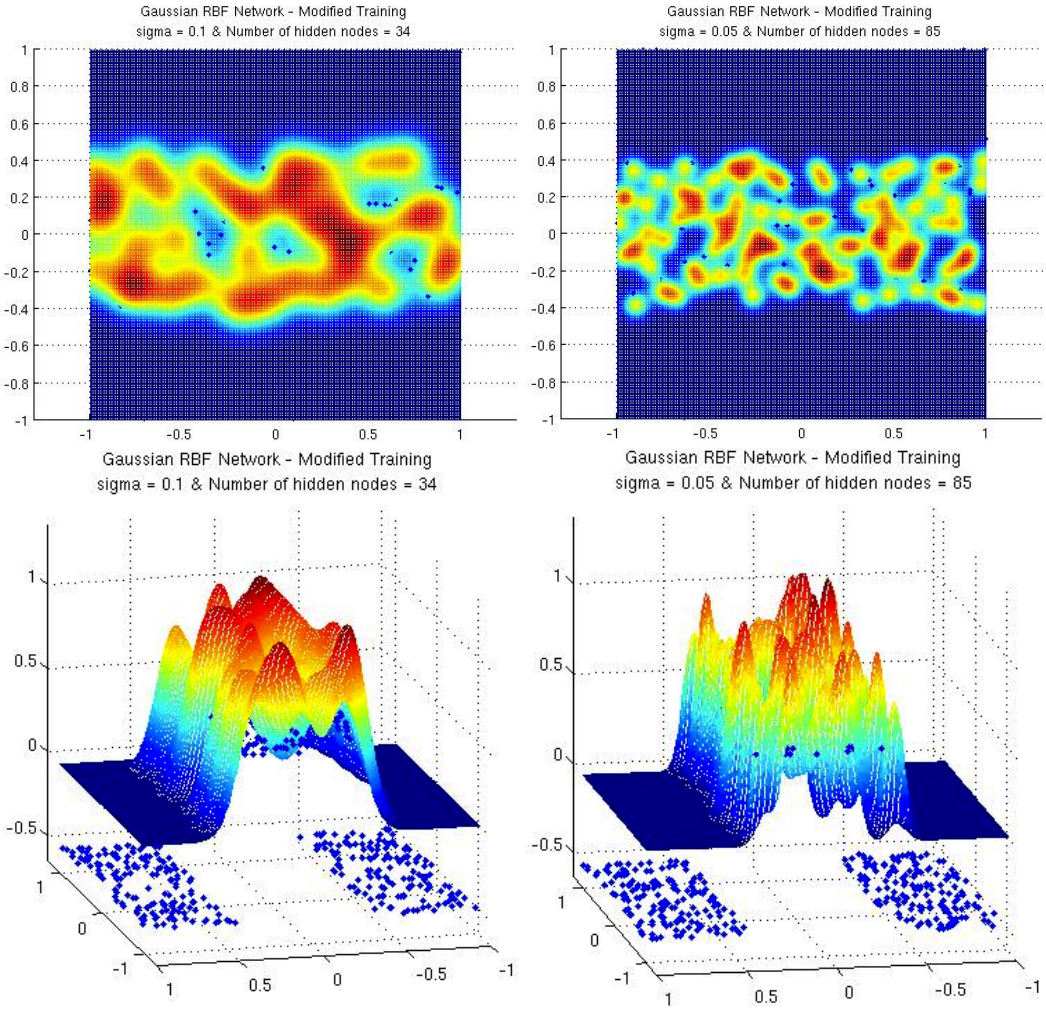


Figure 3.9: Plateau, a 2D function example for estimation.

To evaluate the effect of σ value over the performance of Gaussian RBF networks, it has been adjusted with two different values 0.1 & 0.05. When a higher value has been assigned to standard deviation parameter ($\sigma = 0.1$), Gaussian functions have become more expanded and less number of Gaussian functions have been recruited in hidden layer (34 nodes), and the estimation is more generalized and comprehensive covering more areas around borders. Results of this example is presented in Fig. 3.10a. On the other hand when standard deviation is decreased to $\sigma = 0.05$, number of Gaussian functions recruited in hidden layer increased up to 85 nodes. In this case Gaussian functions are contracted and less generality is achieved as illustrated in Fig. 3.10b. The effect of this phenomenon on classifiers based on Gaussian RBF networks would be on those *False Positive* and *False Negative* errors mentioned in section 3.3. False Negative is expected to decrease when

higher value is assigned to standard deviation parameter ($\sigma \uparrow$), and at the same moment False Positive is supposed to increase. This means with higher value of σ, chances of multi-labeling for each object increases.



(a) $\sigma = 0.1$ (b) $\sigma = 0.05$

Figure 3.10: Gaussian RBF networks estimating the plateau function presented in Fig. 3.9 with two different σ values.

3.5 Novel Learning Algorithm for Gaussian RBF Networks

Purpose of Gaussian RBF Networks (section 3.2) is estimating points distributions in different feature spaces. For such an objective conventional network structure and learning algorithm mentioned before (section 3.4) have a weakness; *same standard deviation in all directions*. This will cause less accuracy in estimation and at the same time a bigger network in terms of hidden nodes. To overcome this problem a novel learning algorithm is proposed in this section. Corresponding ANN Architecture to this novel learning algorithm will be presented in section 3.6.

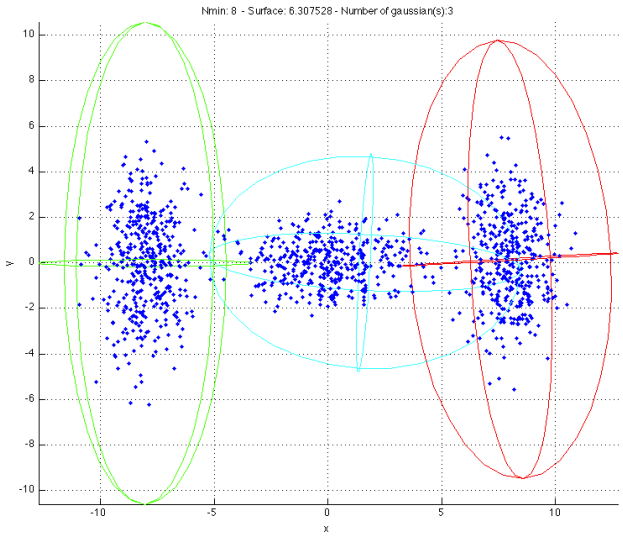


Figure 3.11: Demonstration of Novel learning algorithm.

In conventional learning algorithm inputs are introduced to network and based on the error in the output weights of the network will be adjusted. Unlike the conventional algorithm, the novel algorithm first calculates some Gaussian functions estimating the distribution and then the parameters from obtained Gaussian functions will be used for adjusting the network structure and weights. In other word conventional algorithm is *input-driven*, while the novel algorithm is *output-driven*. The other difference between these algorithm is while the conventional algorithm is iterative, the novel algorithm is recursive.

The key idea is to estimate the whole distribution by a single Gaussian function with help of SVD calculation (the way used for PCA calculation in Appendix A). With respect to 1st principle component obtained via SVD, the distribution will be split in half and each will be estimated with a new Gaussian function. Recursive steps of splitting will continue until the splits are valid. Validity of splits will go false if one of following criteria is reached:

- Number of points covered by each Gaussian function be less than a threshold.
- The split must is not profitable. It means summation of covered area of both two sub-Gaussian functions be less than the primary Gaussian function.

Novel algorithm is implemented by two functions presented in Algorithm 3 and Algorithm 4.

Function *FitGauss* attempts to estimate a set of points with a Gaussian function. For such a purpose it first finds a proper translation (*Gauss.CoG*: Center of Gravity) and rotation matrix (*Gauss.Rotation_Matrix*). Then standard deviations in different directions will be stored (*Gauss.Standard_Deviation*).

Algorithm 3
FUNCTION: FitGauss

INPUT = *Distribution*: a set of points.

$Gauss.CoG$ = center of gravity of the *Distribution*.
move *Distribution* to origin.
$[U, S, V] = SingularValueDecomposition(Distribution)$
$Gauss.Standard\_Deviation = \frac{S}{\sqrt{numberofpoints}}$.
$Gauss.Rotation\_Matrix = V$.

RETURN Gauss

Function *SplitAndMerge* is recursive, and performs the splitting process while checking for validity of the splits. *Distribution* will be split into two *Sub_Distribution*s and the *SplitAndMerge* function will be called for each of them if the split is valid, otherwise the split will be ignored. The split will be executed considering 1st principle component of the *Distribution* obtained via *FitGauss* function stored in *Gauss.Rotation_Matrix* and *Gauss.CoG*. Position of splitting the *Distribution* is where first principle component equals

to zero. When the split is done two criteria of split validity will be examined. As mentioned criteria are based on volume and number of points in each *Sub_Distribution*, where determinant of standard deviation matrix (*Gauss.Standard_Deviation*) some how represents the volume of each Gaussian function. Those outputs of *FitGauss* function called *Gauss* will be accumulated through recursive split.

A demonstration of Gaussian RBF Networks applied to a 3D points distributions is illustrated in Fig. 3.11 and Fig. 3.12.

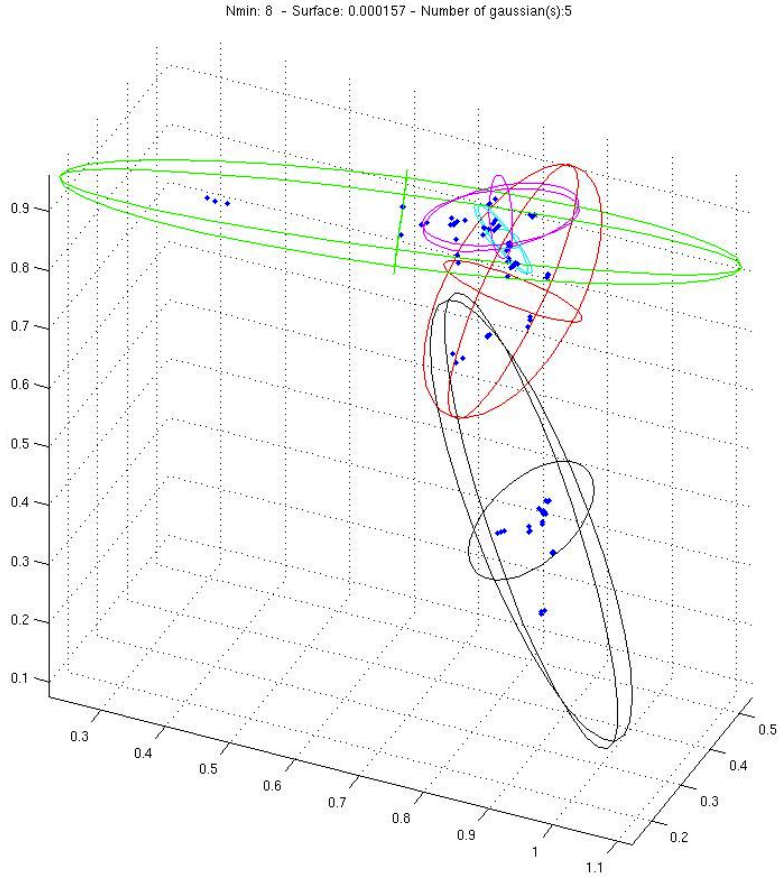


Figure 3.12: Gaussian RBF Networks trained with Novel learning algorithm.

Algorithm 4
FUNCTION: SplitAndMerge

INPUT = *Distribution*: a set of points.
INPUT = *Threshold*: minimum number of points in each *Sub_Distribution*.
INPUT = *Gaussians*: (Null for first call) storage of *Gauss*.

$MainGaus = FitGauss(Distribution)$
with help of *MainGauss*:
Dist_adjusted = move *Distribution* to origin.
$Dist\_adjusted = Dist\_adjusted \times MainGaus.Rotation\_Matrix$

split points of *Distribution* into *SubDist1* and *SubDist2*, regarding wether if first coordinates of correspondent points in *Dist_adjusted* are less or greater than 0.

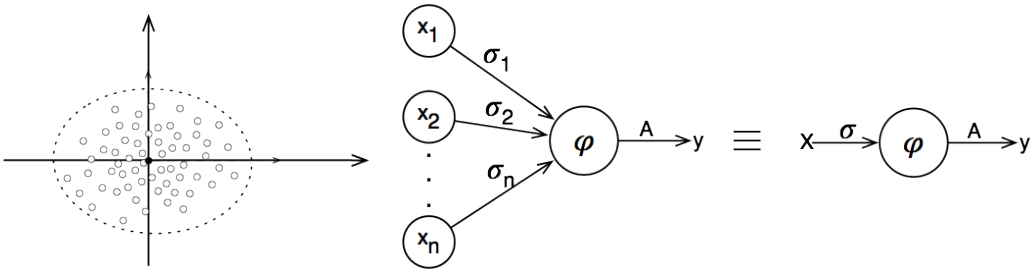
$Volume = determinant(MainGaus.Standard\_Deviation)$;

if $(size(SubDist1)$ *and* $size(SubDist2)) \leq Threshold$ **then**
  add *MainGaus* to *Gaussians*
else
  $GaussiansBackup = Gaussians$;
  $[Gaussians, SubVol1] = SplitAndMerge(SubDist1, Threshold, Gaussians)$
  $[Gaussians, SubVol2] = SplitAndMerge(SubDist2, Threshold, Gaussians)$
  $SubVolume = SubVol1 + SubVol2$
  if $SubVolume > Volume$ **then**
    $Gaussians = GaussiansBackup$
    add *MainGaus* to *Gaussians*
  else
    $Volume = SubVolume$
  end if
end if

RETURN Gaussians and *Volume*

3.6 ANN Architecture for Novel Learning Algorithm

The conventional architecture of RBF network does not have the capability of carrying out the rotation matrix and different standard deviation needed for novel learning algorithm. Therefore a new architecture is proposed here. This Architecture is a combination of two networks, one for translating and rotating the distribution of points to center and aligning with axes of base frame, and second one for realizing a Gaussian function with different standard deviation in different directions.



(a) Gaussian function (b) ANN Architecture

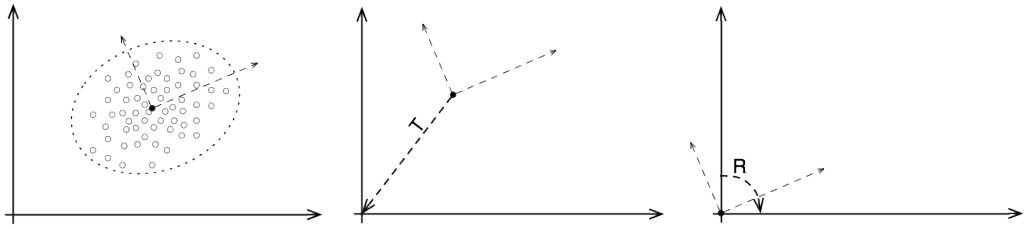
Figure 3.13: Single Gaussian function centered at origin with no orientation, and corresponding ANN network.

Considering a single Gaussian function fitting a distribution demonstrated in Fig. 3.13a located at center and no orientation (equation 3.7), this function could be implemented by a single neuron (Fig. 3.13b), in which input weights are vectors. Mathematical expression of this neuron is given in equation 3.8.

$$y = A \times \exp\left(-\sum_{i=1}^{n}\left(\frac{x_i^2}{2\sigma_i^2}\right)\right) \tag{3.7}$$

$$y = A \times \varphi(X) \quad where \quad \varphi(X) = \exp\left(-\sum_{i=1}^{n}\left(\frac{x_i^2}{2\sigma_i^2}\right)\right) \tag{3.8}$$

If the Gaussian function is not centered at origin and has an orientation, it is still possible to describe it with previous equations (3.8 & 3.7) if the input points are first translated and rotated with same translation and rotation parameters that would bring the Gaussian function to origin with no orientation.



(a) Gaussian function (b) Translation and Rotation

Figure 3.14: Oriented and not centered Gaussian function.

An input vector of distributed points to be evaluated by such a Gaussian function shall be translated and rotated as in equation 3.9, and then simple form of Gaussian equation is applicable for it.

$$\hat{X} = \begin{pmatrix} & R & & T \\ 0 & \cdots & 0 & 1 \end{pmatrix} \times X \quad \rightarrow \quad y = A \times \exp\left(-\sum_{i=1}^{n}\left(\frac{\hat{x}_i^2}{2\sigma_i^2}\right)\right) \tag{3.9}$$

Generalizing the equations for multi dimensional Gaussian function, the translation and rotation could be rewritten as equation 3.10.

$$\hat{X} = (X - C) \times R \tag{3.10}$$

where:

$$\begin{aligned} &C = Center\ of\ gravity\ of\ distribution \\ &R = Rotation\ Matrix \\ &R = \begin{pmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n,1} & r_{n,2} & \cdots & r_{n,n} \end{pmatrix} \qquad C = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} \end{aligned} \tag{3.11}$$

Implementation of translation and rotation with in a single ANN network is illustrated in Fig. 3.15 and described in equation 3.12.

$$\begin{aligned} f_i &= x_i - c_i \\ g_i &= \sum_{j=1}^{n} f_j r_{ji} = \sum_{j=1}^{n} (x_j - c_j)\, r_{ji} \end{aligned} \tag{3.12}$$

Until this point two independent networks have been developed. One for realizing a Gaussian function centered at origin with no orientation, but

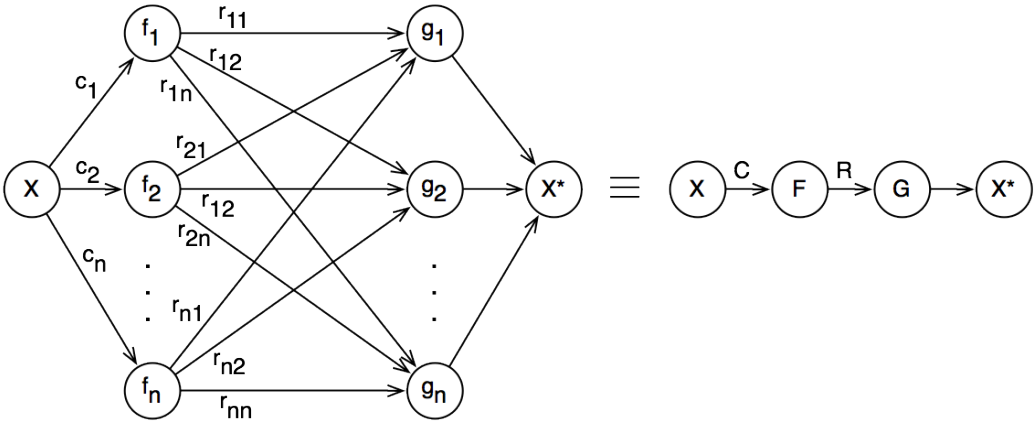


Figure 3.15: ANN Implementation of Translation and Rotation.

with different standard deviation (Fig. 3.13b and equation 3.8). The other one to bring the set of points (distribution) to origin and align its principle components with axes of base frame (Fig. 3.15 and equations 3.10 & 3.12). All parameters needed to adjust these networks would be obtained via Novel learning algorithm developed in section 3.5. Last step would be combination of these two networks. This combination is illustrated in Fig .3.16 and finalized in equation 3.13.

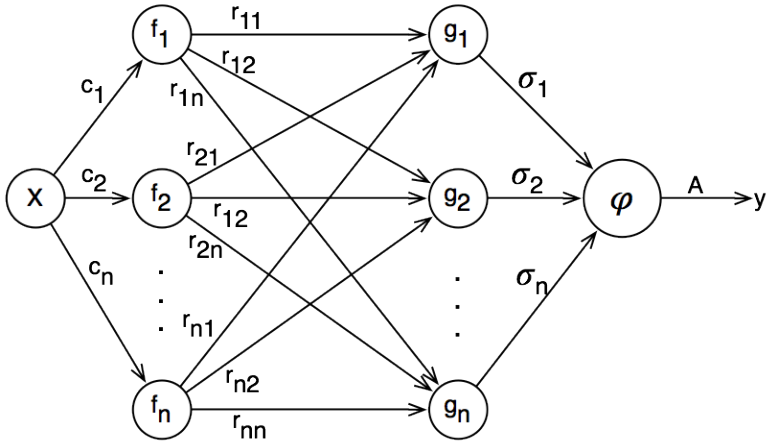


Figure 3.16: ANN Implementation of any arbitrary Gaussian function.

$$y = A \times \exp\left(-\sum_{i=1}^{n}\left(\frac{\left(\sum_{j=1}^{n}\left(x_j - c_j\right) r_{ji}\right)^2}{2\sigma_i^2}\right)\right) \tag{3.13}$$

It should be mentioned that a single network presented in Fig. 3.16 will carry out only one Gaussian function. Hence for estimating a distribution of points with several Gaussian function obtained via *novel learning algorithm*, a set of networks like Fig. 3.16 should work in parallel.

4

CLASS-O-MATIC

Based on requirements of the CART-O-MATIC project, a classifier was called for which could be capable of multi-classification with absolutely one label assigned to each input object. This is unlike essence of the classifier developed based on ANN presented in chapter 3. The classifier was demanded to be most accurate, simple to adjust and fast in practice. Therefore a new classifier called *Classomatic* is presented in this chapter. This classifier could be developed still with supplementary improvements.

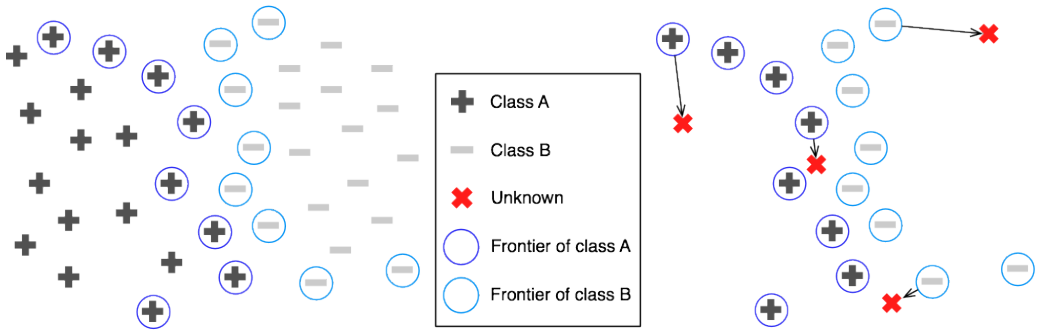
Further than simplicity in adjustment there is another advantage that makes this classifier more competitive. SVM or other conventional classifiers need to be trained for each class separately in case of multi-classification problems as they need *negative reinforcement*. Scenarios of *One versus One* and *One versus All* are used often where a classifier should be trained for each class. Unlike mentioned scenarios, this classifier work in a single module for multi-classification problems with a single training phase.

In first section the idea of this new classifier is presented, then some details of design is presented in section 4.2. Finally in section 4.3 some techniques are assessed for a not completely fulfilled issue of finding frontier points of each class.

4.1 Classification Methodology

The main idea of this classifier is lied in *frontier point*s of all classes in a 12 dimension space including all 4 feature vectors combined. Through the training phase (methods will be discussed in section 4.3), those training

points realizing the frontiers of a class against other classes are detected and stored, rest of the points will be ignored. These *frontier point*s are supposedly sufficient to represent the whole class. Fig. 4.1a demonstrates a presumption of frontier points in a simple 2 classes scenario.



(a) frontier detection (b) classification

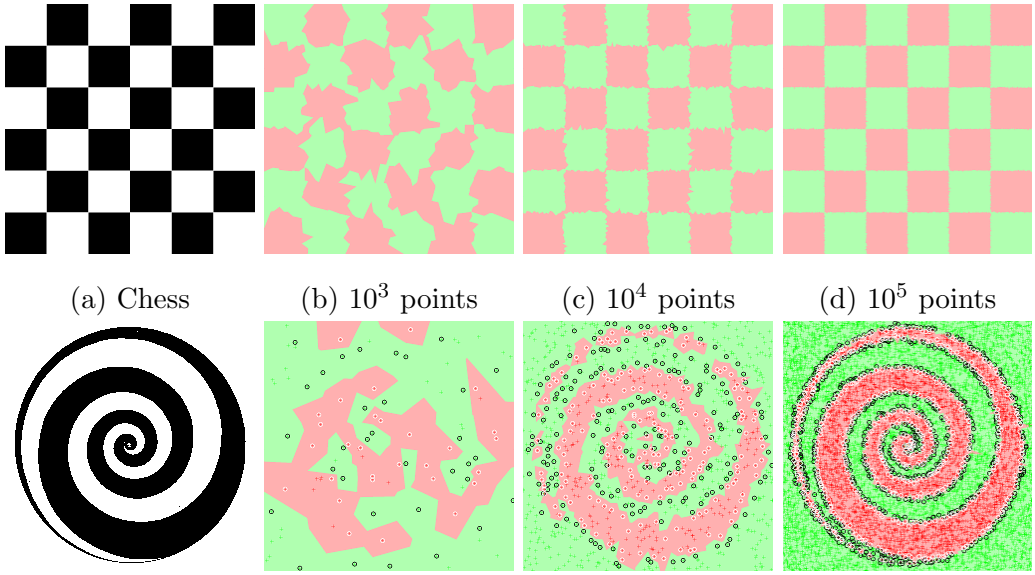
Figure 4.1: Illustration of two synthetic classes, assumed frontier points and classification.

After training phase, when an unknown point arrives distances between input point and all *frontier point*s will be calculated and label of closest one will be assigned to input point(Fig. 4.1b).

While this method executes a multi-class classifier, it potentially can perform a multi-value function estimation (a function with finite and discrete output values). Observing function estimations in Fig. 4.2, dependency of performance on number of training points is quite obvious.

4.2 Distance Calculation & Covariance Matrix

Classomatic is based on closest neighbor in 12 dimension space where dimensions are independent. Hence normalization and distance calculation methods play important roles in performance. Normalization is performed by calculation covariance matrix based on training dataset. Covariance matrix could be based on the whole dataset or separately for each class in dataset, leading to two strategies. On the other hand closeness might be based on



(a) Chess (b) 10^3 points (c) 10^4 points (d) 10^5 points

(e) Spiral (f) 100 points (g) 10^3 points (h) 10^4 points

Figure 4.2: *Classomatic* method employed in function estimation, observing effect of increasing number of training points on performance of *Classomatic*.

Euclidian or Mahalanobis distance. Equation 4.1 is used for distance calculation, in which S is covariance matrix.

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1}(\vec{x}-\vec{y})}$$
$$d(\vec{x},\vec{y}) = \begin{cases} \text{Mahalanobis} & \text{if } S \text{ is full rank} \\ \text{Normalized Euclidian} & \text{if } S \text{ is diagonal} \\ \text{Euclidian} & \text{if } S \text{ is identity} \end{cases} \tag{4.1}$$

And finally for a better assessment feature vectors are supplemented with splitting technique (section 2.4), where a weighting strategy could be employed or not. Based on different options eight possible configurations presented in TABLE 4.1 will be considered and compared to find the optimum adjustment.

Fig. 4.3 demonstrates performance of eight different configurations in term of errors down to 4 level of splitting, applied on those training and testing datasets presented in section 5.2. Fig. 4.3a plots the result of configurations based on Euclidian distance, while the result of Mahalanobis based configurations are plotted in Fig. 4.3b.

	Distance	Covariance Matrix	Sub-Part Weighting
1	Mahalanobis	General	✓
2	Mahalanobis	General	×
3	Mahalanobis	Class-based	✓
4	Mahalanobis	Class-based	×
5	Euclidian	General	✓
6	Euclidian	General	×
7	Euclidian	Class-based	✓
8	Euclidian	Class-based	×

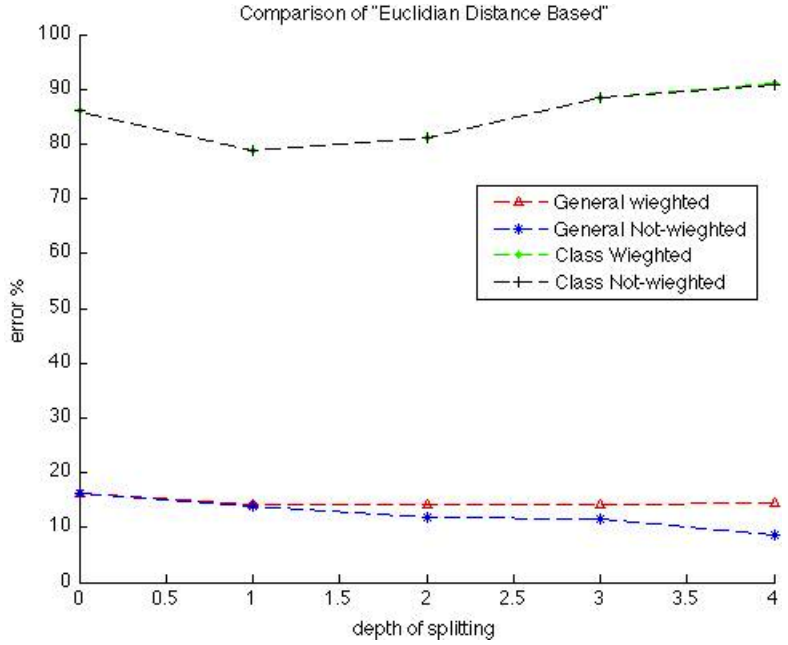
Table 4.1: Eight Configurations of Classomatic.

Observation of results in Fig. 4.3 concludes that *class-based* covariance matrix does not perform properly. Considering ten classes of objects in training dataset and an unknown input object, calculation of distance between input object and objects from ten classes will be done. If distance calculation is based on *class-based* covariance matrix, it would be wrong for nine classes (those which the input object does not belong to) and only matches for one class, while *general* covariance matrix implies to all classes in dataset. Therefore only Mahalanobis distances between input object and objects in the same class make sense, but not with other classes.

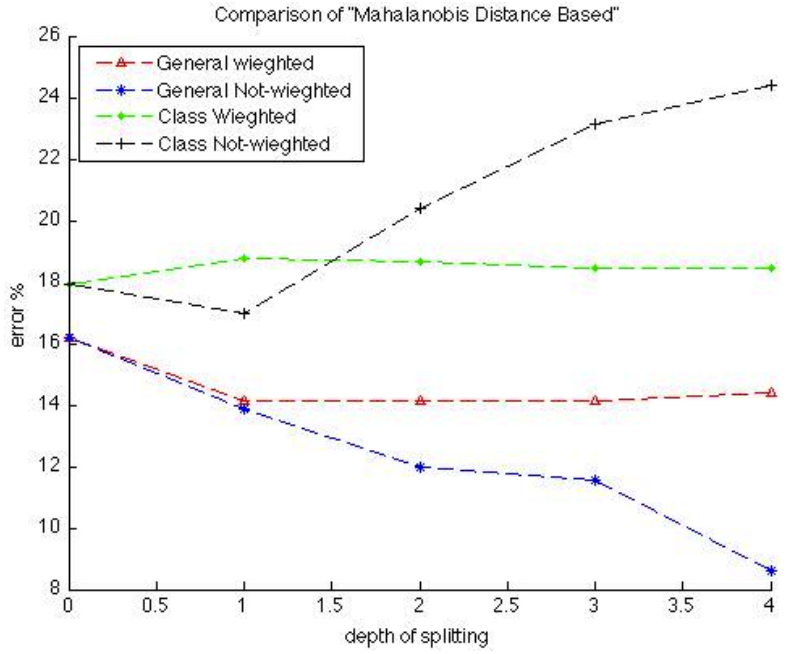
It can be seen in Fig. 4.3 that the best configurations of both Euclidian and Mahalanobis are those with no weighting strategy and covariance matrix calculated from all points of training dataset (*general*). Those two configurations are compared in Fig. 4.4.

When covariance matrix is calculated based on whole dataset (*general*) it's close to a diagonal matrix (no dependency between different dimensions of input), so the results are similar in Mahalanobis and Euclidian when they employ general covariance matrix (Fig. 4.4.).

It should be noted, since the problem of *frontier detection* (and training phase as a consequence) is not completely resolved for a general problem, those experiments during this section for designing process were based on the whole training dataset not only the frontiers. This had not any negative effects on configurations, since frontiers are a subset of training dataset.



(a) euclidian based



(b) mahalanobis based

Figure 4.3: Comparison of *Classomatic*'s eight configuration.

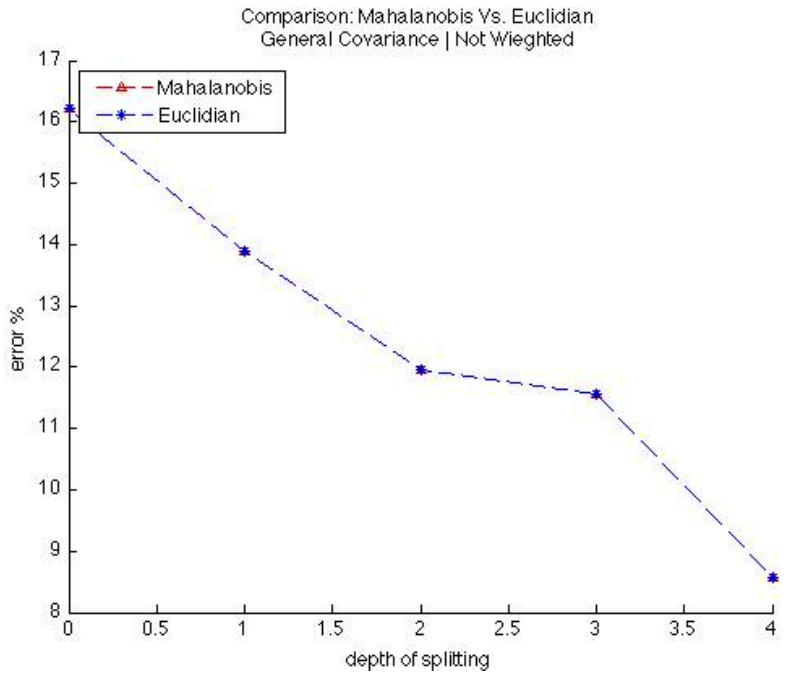


Figure 4.4: Comparing best configurations of Mahalanobis and Euclidian based (General covariance and no weighting strategy).

4.3 Frontier Detection

Novelty of *Classomatic* is based on the idea of representing the training database by *frontier points* of different classes, hence process of *frontier detection* is very important. In this section few techniques will be reviewed, each with some advantages and some drawbacks:

1. Closet dissimilar neighbor,
2. Voronoi tessellation and Triangulation,
3. Support Vectors.

Closet dissimilar neighbor The idea is presented in algorithm 5. To detect if a point (called a) belongs to frontier or not, closest point from any other classes (dissimilar point) to point a will be picked and called b. If closest dissimilar point to b is a itself then a would be a frontier point.

This method is the most easiest one, but does not satisfy the requirements and misses some points of frontier in complicated cases. Therefore an

Algorithm 5 Closet dissimilar neighbor *frontier derection*

Target: detecting frontier points of $CLASS_i$
for all $point_a \in CLASS_i$ **do**
  find closest $point_b$ to $point_a$ where $\{point_b \notin CLASS_i\}$.
  find closest $point_c$ to $point_b$ where $\{point_c \in CLASS_i\}$.
  if $(point_a == point_c)$ **then**
    add $point_a$ to *frontier* set.
  end if
end for

improvement have been done. After each execution of this technique an erosion operation will remove detected *frontier poin*s and the technique will find another layer of *frontier point*s .Fig .4.5 demonstrate result of this technique in practice.

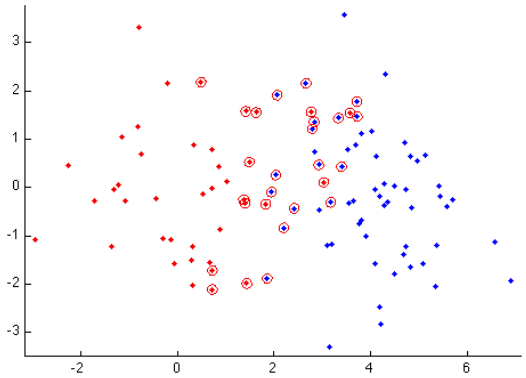
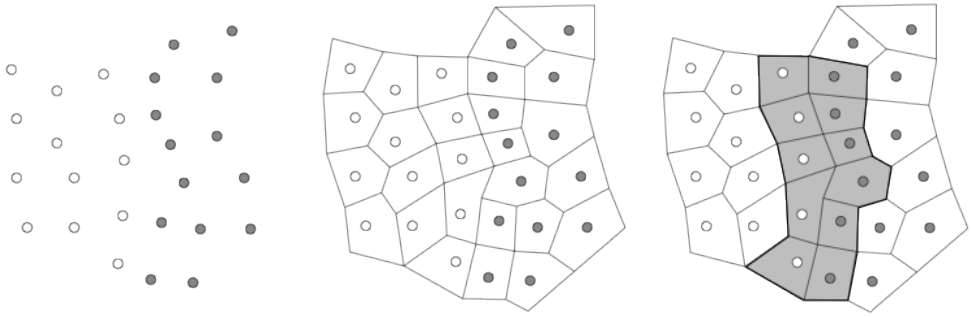


Figure 4.5: Detecting *frontier point*s based on closest dissimilar neighbor, with two times erosion.

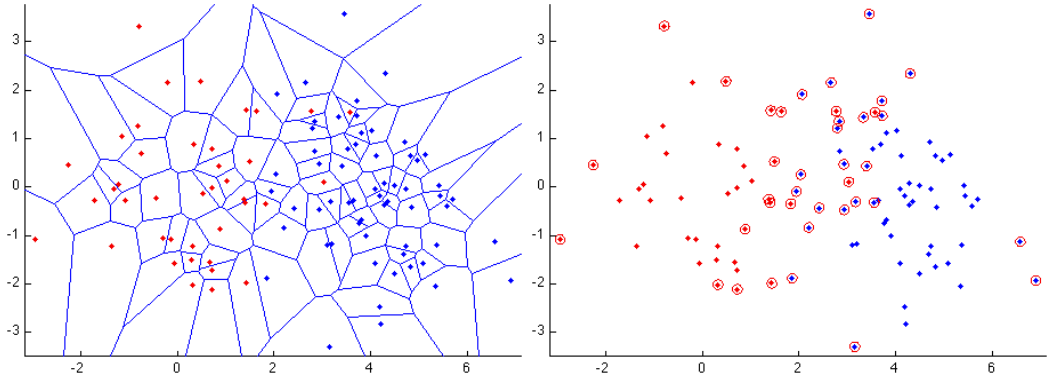
Voronoi tessellation and Triangulation The Voronoi tessellation technique is based on performing a discretization of space in order to define neighborhood for each point. Those points neighboring other classes are flagged as frontier points. The idea is illustrated in Fig. 4.6 based on two synthetic distributions.

Fig. 4.7a demonstrate a Voronoi tessellation, and Fig. 4.7b shows points detected as *frontier*.



(a) synthetic points (b) tessellation (c) frontier points

Figure 4.6: Discretizating a space with Voronoi for *frontier detection*

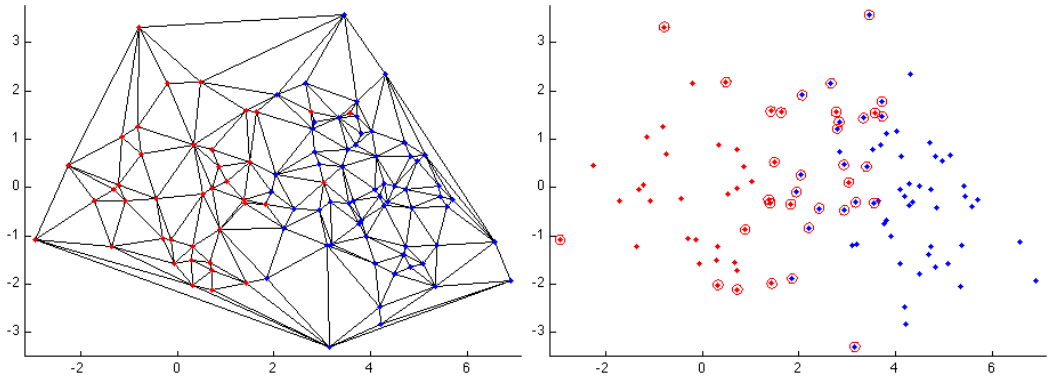


(a) tessellation (b) frontier points

Figure 4.7: Detecting *frontier point*s with Voronoi tessellation.

Triangulation works in a very similar way. Instead of discretization of the space like Voronoi, it would connect points which is the meaning of neighborhood itself. Fig. 4.8a demonstrate a Triangulation and Fig. 4.8b shows detected frontier points with this technique. Based on observations, those *frontier point*s detected by triangulation are always a subset of detected *frontier point*s by Voronoi.

The main problem with Voronoi tessellation and Triangulation techniques is their memory consumption. Required memory for these techniques is an exponential function of space's dimension. For example Voronoi requires $O(n^{\lceil d/2 \rceil})$ storage space[21], where n is number of points and d is dimension of space. This problem obstruct the employment of these techniques for *frontier detection* in case of 12 dimension space (as feature space of this project



(a) triangulation (b) frontier points

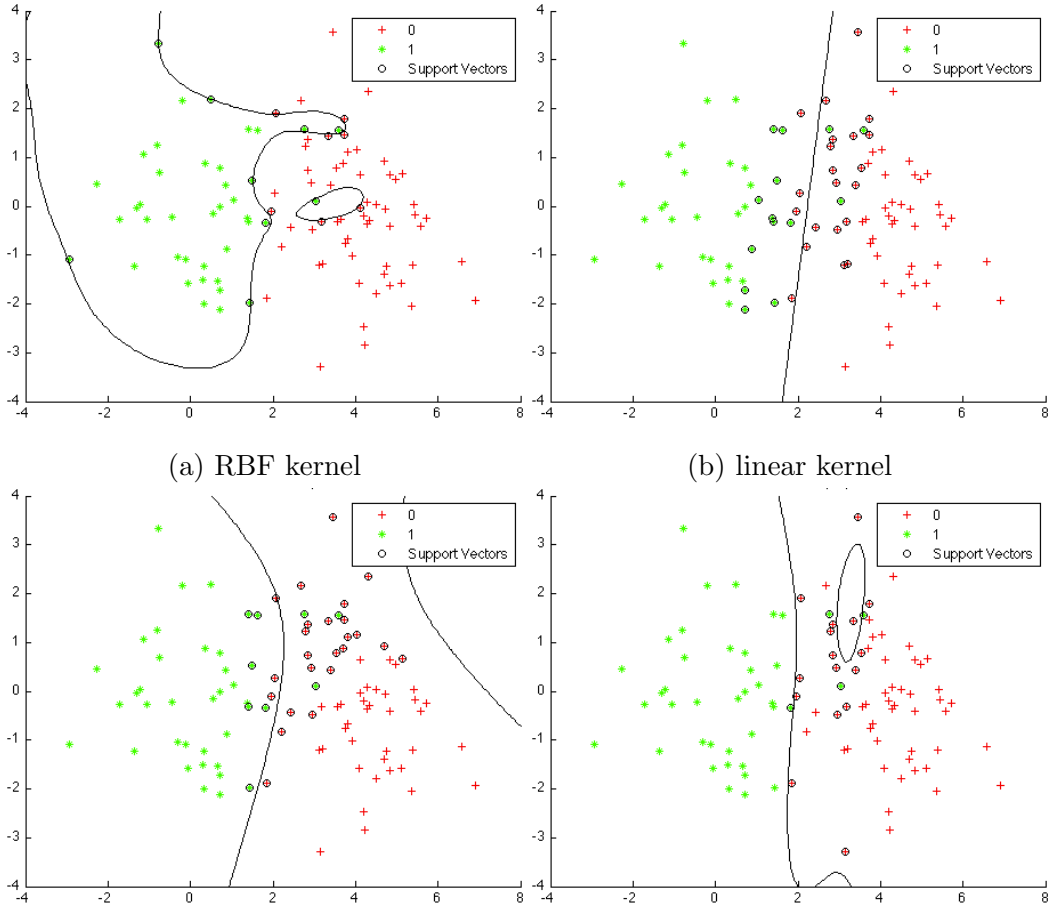
Figure 4.8: Detecting *frontier point*s with Triangulation.

are).

Support Vectors The idea of *SVM* classifiers are to detect *frontier point*s of classes called Support Vectors, and separate them (consequently classes) with a line. If classes are not separable with a straight line in original space, kernel functions would be employed to translate them to a higher dimension space where they are separable[23]. While they are separable with a line (or the function describing the line), any input point could be evaluated if it belongs to one of the classes or the other one. This means any point shall be compared with a line separating two classes and for a multi-class classification strategies like *one versus all* shall be used.

For *frontier detection* of Classomatic, the translation to higher space technique and kernel functions are adopted from SVM. When *frontier point*s set is constructed with this technique, there is no need to set up a line/function with margins or other parameters of SVM, but unknown input point would be compared simply with all points in the *frontier point*s set. Fig. 4.9 demonstrates *frontier point*s or equally support vectors of two classes, where four different kernel functions were employed.

As Fig. 4.9 shows *frontier detection* in sense of Classomatic application is not performed perfectly, but the advantages of this technique is that it can be employed for high dimension spaces and there exist several options for kernel function.



(a) RBF kernel

(b) linear kernel

(c) quadratic kernel

(d) polynomial kernel

Figure 4.9: SVM employed for *frontier detection* with different kernel functions

5

Experimental Results

Purpose of this master's thesis was to develop classification methods for integrating into *CARTOMATIC* project participating in *CAROTTE* competition (presented in introduction chapter). Hence it would be better to evaluate the performance of developed methods with a database acquired from the competition context. First the robotic platform designed and realized by CARTOMATIC in Angers will be presented. It is called *MiniRex* and it has been employed for constructing the databases. In next section (5.2) the databases will be presented. There will be two other sections (5.3 & 5.4) presenting the performance of two classifiers, one based on ANN presented in chapter 3, and the other one *Classomatic* presented in chapter 4. In section 5.3 performance of ANN-based classifier is assessed by comparing two different learning algorithms, while in section 5.4 a comparison between *Classomatic* and *SVM* classifiers is carried out.

5.1 MiniRex the Robotic Platform

The aim of this contest was to map and locate objects in a structured environment similar to an apartment. The particularity of CARTOMATIC team was the use of a multi-robot strategy [2]. During the navigation, an object is usually seen from several viewpoints by many robots (Fig. 5.2). This also justify choice of a classifier that can provide several candidates as output, by crossing information from each robot, ambiguities can usually be solved. CARTOMATIC team designed and built seven identical mobile robots called *MiniRex* (MINIature Robot for Exploration) illustrated in Fig. 5.1. Each robot is composed of an Embedded PC (proc. Atom 1.6GHz), inclinometer and ultrasonic sensors for navigation, LIDAR for localization and mapping,

Figure 5.1: MiniRex; the platform

and an RGB-D sensor (Microsoft Kinect) for object recognition. The built-in actuator of the Kinect has been locked and the sensor was calibrated with a new reference frame merged with the robot's coordinate system.

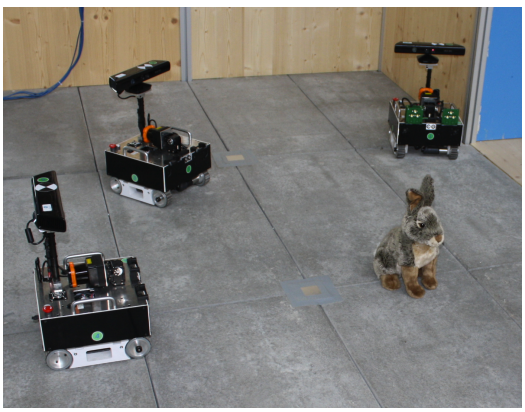


Figure 5.2: Acquisition with several MiniRex

5.2 Database

Ten classes of objects (Fig. 5.3) are picked out of those proposed by *Défi-CAROTTE*. Initially a very small database was constructed by capturing objects with the robots as illustrated on Fig. 5.2. Eventually the database grew bigger by appending those captures from several robotic missions launched during the process of development, that is to say the current version of the database contain real case data and the results are very reliable.

Figure 5.3: 10 classes of objects picked for experiments

The database was randomly distributed into *Training* and *Testing* datasets, delivering more to the training set. TABLE 5.1 presents all classes as well numerating the objects included.

5.3 Results on Gaussian RBF networks classifiers

In order to employ the ANN based classifier, some parameters have to be adjusted. While two different learning algorithm don't share any parameter, their adjustment will be independent. For each learning algorithm two configurations will be proposed and employed (sub-sections 5.3.1 & 5.3.2). Results of two different learning algorithm will be compared in one single table (TABLE 5.6). Since this classifier performs a multi-labeling strategy, results are presented with two parameters:

	Number of Items		
Class	Total	Training	Testing
Black Chair	57	31	26
Wheeled Chair	170	116	54
Yellow Chair	121	86	35
White Box	94	65	29
Blue Box	62	39	23
Orange Cylinder	87	50	37
Green Cylinder	71	42	29
Fan	85	50	35
Red Ball	74	39	35
Plant	85	43	42
OVERALL	906	561	345

Table 5.1: DataBase

- *CR: Correct Recognition* represents those objects assigned with correct label,
- *FP: False Positive* represents number of correctly recognized objects with multi-labels.

5.3.1 Settings parameters for conventional learning algorithm

As mentioned in the section 3.4 the parameter σ should be close to the "half of the largest distance between nearest neighbors"[29]. Based on this hypothesis estimated σ for each network comes in TABLE 5.2. In this experiment due to inadequate number of training points and non-exhaustive dataset, σ will be a multiplication of the estimated value in order to expand the distribution's estimation. In configuration (TABLE 5.3) a coefficient to these σ values will be adjusted. Size of an object in any dimension varies from 2.0×10^{-4} to 1.05×10^{3} with an average of 327.2, similarly variance of mass varies from 7.0×10^{-5} to 2.83×10^{2} with an average of 77.5, and limitations of color space would be $0^o < H < 360^o$, $0 < S < 1$ and $0 < V < 1$.

In addition to σ estimation *Bugmann* proposed values for those variables mentioned in Algorithm 2 (*learnrate*, *threshold*, *tolerance* and Number of *"epochs"*) [29]. Based on experiments turned out only the σ affects the performance of the classifier critically, therefore performance of classifier has

	Estimated σ in distinctive networks			
	Size	Mass PCA	Color	Color PCA
Black Chair	50.00	15.49	4.03	4.22
Wheeled Chair	50.00	20.28	3.34	3.96
Yellow Chair	50.00	20.11	3.92	2.51
White Box	35.71	8.68	15.22	14.51
Blue Box	25.71	3.75	5.42	5.14
Orange Cylinder	39.78	7.56	7.25	8.83
Green Cylinder	11.24	2.67	10.41	10.46
Fan	49.38	16.10	13.40	13.09
Red Ball	12.89	3.77	15.10	13.45
Plant	23.13	6.50	2.76	3.38

Table 5.2: σ Estimation

been assessed by observing the errors while all parameters except the coefficient of σ were adjusted to those values of TABLE 5.3 and coefficient of σ was variating from 0.2 to 8.

	1*st* Configuration	2*nd* Configuration
Coefficient of σ	2.1	8
Tolerance	0.15	
Number of "Epoch"	5	
Learning rate	0.51	
Threshold	$0.5\times\sigma$	

Table 5.3: RBF Network Configurations

Proposed configurations in TABLE 5.3 have been applied to the classifier and the corresponding results are presented in TABLE 5.4.

TABLE 5.4 shows that with increment of σ value, estimation of distributions become more dominant. This will lead to better results in term of *Correct Recognition* with price of increasing *False Positive* error of multi-labeling.

		1*st* Configuration		2*nd* Configuration	
Class	Items	C R	F P	C R	F P
Black Chair	26	8	4	17	17
Wheeled Chair	54	23	12	43	34
Yellow Chair	35	20	0	30	7
White Box	29	22	17	26	26
Blue Box	23	8	7	18	18
Orange Cylinder	37	28	20	35	35
Green Cylinder	29	23	11	28	28
Fan	35	20	18	34	34
Red Ball	35	29	24	34	34
Plant	42	28	22	39	39
TOTAL	345	209	135	304	272
Correct Recognition (%)		60.57%		88.11%	

Table 5.4: Classification results of ANN based classifier with conventional learning algorithm.

5.3.2 Settings parameters for novel learning algorithm

Novel learning algorithm presented in section 3.5 has only two parameters to adjust:

- Minimum number of points allowed in each single Gaussian function. This would be one of the criteria of stopping the recursion.
- Threshold of membership in each Gaussian function. If the output of a Gaussian function is above this threshold, the point is included in that function. This value will be a coefficient of the σ value.

Like conventional learning algorithm, two configurations are proposed for the novel learning algorithm. Based on experiments minimum number of points in each gaussian function is supposed to be optimum by *10 points*. Therefore it would be fixed as 10 while the value of thresholding the output of gaussian function would vary from $e^{\frac{-4.5^2}{2}}$ in 1st configuration to $e^{\frac{-5.5^2}{2}}$ in 2nd configuration. TABLE. 5.5 presents the results of ANN based classifier trained with novel learning algorithm.

Similarly to conventional learning algorithm change of σ value, cause a change estimation's dominancy. That is to say with increasing σ value *Correct Recognition* will improve and *False Positive* error of multi-labeling

		1*st* Configuration		2*nd* Configuration	
Class	Items	C R	F P	C R	F P
Black Chair	26	23	15	26	22
Wheeled Chair	54	52	12	54	20
Yellow Chair	35	25	0	26	0
White Box	29	29	27	29	29
Blue Box	23	20	17	20	20
Orange Cylinder	37	33	0	34	0
Green Cylinder	29	19	11	23	17
Fan	35	35	23	35	30
Red Ball	35	30	1	32	2
Plant	42	41	1	42	2
TOTAL	345	307	107	321	142
Correct Recognition (%)		88.98%		93.04%	

Table 5.5: Classification results of ANN based classifier with novel learning algorithm.

increases too.

5.3.3 Comparison of two learning algorithms

Behavior of two learning algorithm is very similar, as their dominancy of estimation expands, *Correct Recognition* improves and multi-labeling increases to. In TABLE. 5.6 those configurations of different learning algorithms are compared that yield similar *Correct Recognition*, so they could be compared in terms of *False Positive* error or multi-labels per object.

As TABLE. 5.6 shows, with similar *Correct Recognition* the *False Positive* error of conventional algorithm is more than two times bigger than the novel algorithm.

5.4 Results on *Classomatic*

Performance of *Classomatic* is assessed compared to a well known conventional classifier, the *SVM*. Based on experiments the best kernel function suited for database of this project, is supposed to be RBFs. That's because the optimization problem of *SVM* would not converge with other kernel func-

		Novel Learning		Conventional Learning	
Class	Items	C R	F P	C R	F P
Black Chair	26	23	15	17	17
Wheeled Chair	54	52	12	43	34
Yellow Chair	35	25	0	30	7
White Box	29	29	27	26	26
Blue Box	23	20	17	18	18
Orange Cylinder	37	33	0	35	35
Green Cylinder	29	19	11	28	28
Fan	35	35	23	34	34
Red Ball	35	30	1	34	34
Plant	42	41	1	39	39
TOTAL	345	307	107	304	272
Correct Recognition (%)		88.98%		88.11%	
False Positive (%)		34.85%		89.47%	

Table 5.6: Result comparison of two learning algorithms employed by ANN based classifier.

tion (even order 40 of polynomial function). Therefore RBF kernel function have been engaged with $\sigma = 1$.

To make the the classifiers more equivalent and have a more reliable comparison, the *SVM* itself have been employed amongst different *frontier detection* techniques for Classomatic (section 4.3). Results of the two classifiers are shown in TABLE 5.7.

As TABLE 5.7 demonstrates, recognition errors of Classomatic (14.79%) is half of the errors with SVM (27.54). Although they both use the same method for detecting *Frontier Point*s/*Support Vectore*s, Classomatic behaves more accurate. The reason is that SVM uses Support Vectors to obtain a line separating different classes, this line is estimated by some functions. This process is not as accurate as using distance to Frontier Points themselves to separate classes.

		Correct Recognition	
Class	Items	Classomatic	SVM
Black Chair	26	13	8
Wheeled Chair	54	49	40
Yellow Chair	35	34	24
White Box	29	18	8
Blue Box	23	14	17
Orange Cylinder	37	34	32
Green Cylinder	29	27	28
Fan	35	30	29
Red Ball	35	35	31
Plant	42	40	33
TOTAL	345	294	250
Correct Recognition (%)		85.21%	72.46%

Table 5.7: Comparison between *Classomatic* and *SVM*.

6

Conclusions

An object recognition problem has been dealt in this master's thesis integrated in CARTOMATIC project in framework of Carotte competition. The competition was based on environment exploration with autonomous mobile robots. Several objects from different classes are spread in arena and robots should detect and recognize most possible items.

MiniRex, a mobile robotic platform has been designed by CARTOMATIC team with an RGB-D camera embedded for visual acquisitions. Acquired images leads to a $3D$ colored point cloud of the scene, and the arena would be scanned completely with multi robots working in parallel. Acquisition will be analyzed and all regions representing a possible object would be detected and isolated. Next step is extraction of four $3D$ global feature vectors based on size and color of the objects. Those feature sets studied and proposed here have advantage of simplicity in extraction without compromising discriminative characteristic of global features.

For classification two methods were developed, a multi-labeling classifier based on ANN and a single-label classifier:

- ANN based classifier for multi-labeling,
- Classomatic for single-labeling.

Most interesting features of proposed multi-labeling classifier are its mechanism and un-necessity of "negative reinforcement". These features provide classifiers independent from each other and capable of handling uncertainty. Proposed classifier based on ANN as well has advantage of fine tuning carrying out an adaptive system for variety of feature sets and also other applications. Dynamics, high speed and low computational cost of the method

makes it suitable for integration into a more exhaustive recognition system as a pre-classifier in order to decrease computational costs in supplementary accurate recognition stage. After all novel learning algorithm proposed for this classifier exploit all potential of the network toward more accuracy.

Classomatic, the second classification method proposed for single labeling, is interestingly simple in idea and implementation. This simplicity makes it suitable for a variety of applications, without scarifying the accuracy. The comparison conducted between Classomatic and SVM's performance justifies that. Different *frontier detection* techniques also gives the advantage of fitting for diverse problems.

All tests were performed based on a very realistic database containing some objects even not recognizable with human experience. ANN based classification method shows high rate of recognition that could converge to full recognition with the novel learning algorithm, while *False Positive* error would not pass 50%. That means more that 50% of objects would be recognized correctly with only one label, and the rest of them will have more than one label which one of them is the correct one. As well Classomatic demonstrates higher accuracy compared to SVM without higher computational cost.

Although those classifiers proposed in this project are suitable for any application, prospective works following this project in the filed of Computer Vision and Object Recognition would be constructing a general model of a recognition system with supplementary local feature extractions for accurate decision. General control system may include a fuzzy logic decisioning which would demand for further feature extraction (feature vector enrichment) when the outputs of classifiers are not specific (in case of ambiguity). This will lead to more accurate and comprehensive recognition system, while keeps the speed up for distinctive objects who don't need rich feature vectors for recognition.

Appendix A: Using *Singular Value Decomposition* for *PCA* Calculation

Singular Value Decomposition of Matrix X is expressed as:

$$X = U\Sigma V^T$$

where:

- U is an $m \times n$ matrix and its columns are called left singular vectors. Its columns form an orthonormal basis:

$$u_i.u_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

- Σ is an $n \times n$ diagonal matrix and called contains singular values in order of hight o low on diagonal.

- V^T: is an $n \times n$ matrix; and its rows are called right singular vectors. Rows of V^T form an orthonormal basis too.

There exist both numerical and analytical methods for Singular Value Decomposition. It has a variety of applications from *Pseudoinverse* to *Null space* analysis. It has been shown there is precise relation between SVD analysis and Principal Component Analysis [20]. In this Project SVD has been employed for Principle Component Analysis, and it is calculated via integrated $MATLAB^{®}$ function.

If center of gravity of a distribution of points is aligned with origin of the frame, relation between SVD and PCA could be described as:

- Σ is the length of each principle component. In another term, it's the standard deviation of the points distribution along principle components.

- V^T is a rotation matrix from base frame to a frame aligned with principle components.

Fig. 6.1 demonstrates principle components of a Gaussian distribution in $3D$. Red lines are aligned with principle components of the distribution and their length represents the standard deviations.

There exist numerous detailed and comprehensive discussions in literature about SVD and PCA [46, 40, 32, 45, 43] which are not concerned here.

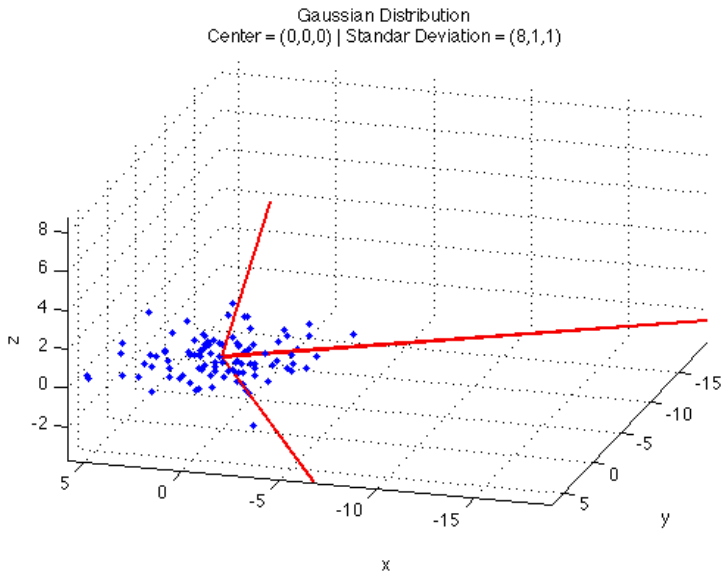


Figure 6.1: PCA of a $3D$ Gaussian distribution.

Bibliography

[1] S. Gholami Shahbandi, P. Lucidarme, *Object Recognition Based on Radial Basis Function Neural Networks: experiments with RGB-D camera embedded on mobile robots*, IEEE 1st International conference on Systems and Computer Science, Lille, France, August 29-31, 2012.

[2] A. Bautin, O. Simonin and F. Charpillet, *Towards a communication free coordination for multi-robot exploration*, CAR 2011, 6th National Conference Control Architecture of Robots, May 2011, Grenoble, France, 8p.

[3] Amitabh Wahi, Priyadharshini Ravi, M. Saranya, *A neural network approach to rotated object recognition based on edge features: Recognition rate and CPU time improvement for rotated object recognition using DWT*, IEEE, International Conference on Computing Communication and Networking Technologies (ICCCNT), July 2010.

[4] Pablo Zegers, *State of the Art in Neural Networks*, Lecture Note, College of Engineering and Applied Sciences, University of the Andes, Chile, 2009.

[5] H. Bay, A. Ess, T. Tuytelaars, L. Gool, *SURF: Speeded Up Robust Features*, Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346-359, 2008.

[6] Peter M. Roth, Martin Winter, *Survey of Appearance-based Methods for Object Recognition*, Technical Report, Graz University of Technology, Inst. f. Computer Graphics and Vision, ICG-TR-01/08, 2008.

[7] L. Wang , J. Shi , G. Song and I-fan Shen *Object Detection Combining Recognition and Segmentation*, ACCV'07 Proceedings of the 8th Asian conference on Computer vision, Part I, pp 189-199, 2007.

[8] Guobin Ou, Yi Lu Murphey, *Multi-class pattern classification using neural networks*, ELSEVIER, Journal on Pattern Recognition, Vol. 40, Issue 1, Pages 4-18, January 2007.

[9] Ming-Hsuan Yang, *Object Recognition*, Encyclopedia of Database Systems (eds. Ling Liu and M. Tamer Ozsu), pp. 1936-1939, 2009.

[10] Ji-Xiang Du, et al, *Shape recognition based on neural networks trained by differential evolution algorithm*, ELSEVIER, Journal on Neurocomputing, Vol. 70, Issues 46, Pages 896903, January 2007.

[11] Grigorios Tsoumakas, Ioannis Katakis. *Multi-Label Classification: An Overview*, International Journal of Data Warehousing & Mining, 3(3), 1-13, July-September 2007.

[12] J. Mutch and D. G. Lowe, *Multiclass Object Recognition with Sparse, Localized Features*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p.11-18, June 17-22, 2006.

[13] Kilian Q. Weinberger, John Blitzer and Lawrence K. Saul, *Distance metric learning for large margin nearest neighbor classification*, In NIPS, MIT Press, 2006.

[14] L. Chen, *Pattern Classification by Assembling Small Neural Networks*, IEEE International Joint Conference on Neural Networks, IJCNN ,vol. 3, pp. 1947-1952, 2005.

[15] Leonardo Noriega, *Multilayer Perceptron Tutorial*, Staffordshire University, 2005.

[16] D. Lowe, *Distinctive image features from scale-invariant keypoints*, INTERNATIONAL JOURNAL OF COMPUTER VISION Volume 60, Number 2 (2004), 91-110, DOI: 10.1023/B:VISI.0000029664.99615.94.

[17] Anna Bartkowiak, *NEURAL NETWORKS and PATTERN RECOGNITION*, Lecture notes, Institute of Computer Science, University of Wroclaw, 2004.

[18] K. Fukushima, *Neural Network Models for Vision*, International Joint Conference on Neural Networks, IEEE, vol. 4, pp. 2625-2630, 2003.

[19] S. Hinz, *Integrating Local and Global Features for Vehicle Detection in High Resolution Arial Imagery*,ISPRS Archives, Vol. XXXIV, Part 3/W8, Munich, 17.-19. Sept. 2003.

[20] Michael E. Wall, Andreas Rechtsteiner, Luis M. Rocha. *Singular value decomposition and principal component analysis*, in A Practical Approach to Microarray Data Analysis. D.P. Berrar, W. Dubitzky, M. Granzow, eds. pp. 91-109, Kluwer: Norwell, MA (2003). LANL LA-UR-02-4001.

[21] S. Arya, T. Malamatos, and D. M. Mount, *Space-Efficient Approximate Voronoi Diagrams*, Proc. 34th ACM Symp. on Theory of Computing (STOC 2002), pp. 721730.

[22] Yann Frauel, Bahram Javidi, *Neural network for three-dimensional object recognition based on digital holography*, Optics Letters, Vol. 26, Issue 19, pp. 1478-1480, 2001.

[23] Cristianini, N. and Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, First Edition (Cambridge: Cambridge University Press), 2000.

[24] G. P. Zhang, *Neural Networks for Classification: A survey*, *IEEE Trans.* Systems, Man and Cybernetics - Part C: Application and Reviews, vol. 30, No. 4, pp 451-462, 2000.

[25] O. Chapelle, P. Haffner, V. N. Vapnik, *Support vector machines for histogram-based image classification*, IEEE Transactions on Neural Networks, vol. 10, Issue. 5, pp. 1055-1064, 1999.

[26] D. J. Moore, I. A. Essa, M .H .Hayes, *Exploiting human actions and object context for recognition tasks*, The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 1 ,pp. 80-86, 1999.

[27] P. N. Suganthan, Eam Khwang Teoh, Dinesh P. Mital, *Hopfield Network with Constraint Parameter Adaptation for Overlapped Shape Recognition*, IEEE Trans. on Neural Networks, Vol. 10, Issue 2, Pages 444-449, Mar 1999.

[28] C R. Shyu, et al, *Local versus Global Features for Content-Based Image Retrieval*, IEEE Workshop on Content-Based Access of Image and Video Libraries, 1998,

[29] G. Bugmann, *Normalized Gaussian Radial Basis Function Networks*, *Neurocomputing* [special issue on Radial Basis Function Networks] vol. 20 , pp 97110, 1998.

[30] B. Krebs, F. M. Wahl, *Automatic generation of Bayesian nets for 3D object recognition*, Fourteenth International Conference on Pattern Recognition, vol. 1, pp. 126-128, 16-20 Aug 1998.

[31] Christopher J.C. Burges, *A Tutorial on Support Vector Machines for Pattern Recognition*, Appeared in Data Mining and Knowledge Discovery Vol. 2, pp. 121-167, 1998.

[32] G. Strang, *Introduction to Linear Algebra*, Wellesley, MA: Wellesley Cambridge Press, 1998.

[33] N. A. Thacker, *Tutorial: Supervised Neural Networks in Machine Vision*, Statistics memo from TINA[1], University of Manchester, Features and Measurement Series (TINA), 1997.

[34] JIANN-SHU LEE, et al, *Occluded Objects Recognition Using Multiscale Features and Hopfield Neural Network*, ELSEVIER , The Journal of Pattern Recognition, Vol. 30, Issue 1, Pages 113122, January 1997.

[35] Ral Rojas, *Neural Networks: A Systematic Introduction*, Book, Springer, 502 p. 154 illus. , ISBN 978-3-540-60505-8, 1996.

[36] JUNG H. KIM, SUNG H. YOON, KWANG H. SOHN, *A Robust Boundary-Based Object Recognition in Occlusion Environment by hybrid Hopfield Neural Networks*, ELSEVIER, The Journal of Pattern Recognition, vol. 29, no. 12, pp 2047-2060, December 1996.

[37] Anil K. Jain and K. M. Mohiuddin, *Artificial Neural Networks: A Tutorial*, IEEE Computational Science & Engineering, Journal Computer, Special issue: neural computing: companion issue to Spring 1996, Volume 29 Issue 3, Page 31-44, March 1996.

[38] E. Osuna, R. Freund, F. Girosit, *Training support vector machines: an application to face detection*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 130-136, 1996.

[39] Du-Ming Tsai, Ray-Yuan Tsai, *Use neural networks to determine matching order for recognizing overlapping objects*, ELSEVIER Journal, Pattern Recognition Letters, Vol. 17, Issue 10, Pages 1077-1088, Sept. 2, 1996.

[40] G. Golub, C. Van Loan, *Matrix Computations. Baltimore* Johns Hopkins Univ Press, 1996.

[41] C. M. Bishop, *Neural networks for pattern recognition*, *Clarendon Press*, Oxford, UK, pp 170, 1995.

[42] A. Ravichandran, B. Yegnanarayana *Studies on Object Recognition From Degraded Images Using Neural Networks*, ELSEVIER Journal, Neural Networks, Vol. 8, No. 3, pp. 481-488, 1995.

[1] http://www.tina-vision.net/index.php

[43] E. R. Jessup, D. C. Sorensen, *A parallel algorithm for computing the singular-value decomposition of a matrix*, Siam Journal on Matrix Analysis and Applications 1994; 15:530-48.

[44] Peter W. M. Tsang, P. C. Yuen, *Recognition of Partially Occluded Objects*,IEEE Trans. System, Man and Cybernetics, vol. 23, no. 1, Jan/Feb 1993.

[45] M. W. Berry, *Large-scale sparse singular value computations*, International Journal of Supercomputer Applications 1992; 6:13-49.

[46] I. T. Jolliffe, *Principal Component Analysis*, New York: Springer, 1986.