FACULTY OF ENGINEERING OF THE UNIVERSITY OF PORTO

U. PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Object Recognition and Pose Estimation in Flexible Robotic Cells

Luís André Freitas da Rocha

Doctoral Program in Electrical and Computer Engineering

Adviser: Prof. Dr. António Paulo Gomes Mendes Moreira

Co-Adviser: Prof. Dr. Vitor Manuel Ferreira dos Santos

May 19, 2014

Abstract

In modern production systems, the main challenge is to meet customers' demands while maintaining products quality and production rate. In this scenario, state-of-art solutions are required which promote an enhancement in the production process flexibility and efficiency. These solutions focus in technological developments that allow set up times to be reduced and increasing efficiency in the operations performance.

Considering flexibility, industrial robots still present some limitations that prevent them to be used in vast fields of industry. These limitations are directly related with the programming flexibility and the trajectory adaptation with the product positioning, that is the target of robot interaction.

Therefore, in this Ph.D. thesis, the main objective is to contribute to solve the problem of lack of industrial robot perception skills, by proposing a robust 3D part recognition and localization system suitable for robotics conveyor lines. These emergent requirements are driven by the needs specified by enterprises with small production series (characterized by a wide range of products produced simultaneously) that seek for full robotic automation of their manufacturing line. Furthermore, their production process characteristics may also limit the use of some conventional identification systems like RFID or bar codes. These limitations are, for example, imposed by heat treatments or painting operations, which can derail the use of these well-known approaches. Plus, the mechanical structure of the conveyor also introduces a geometric inaccuracy in the object positioning.

With the development of an industrial robot complementary system, responsible for object recognition and pose estimation, the industrial robot will be able to select and adjust autonomously the correct program to execute. For this purpose, a cascade system performed with Support Vector Machine and Perfect Match was developed. All the recognition procedure and pose estimation is performed in 3 seconds maximum with standard off the shelf hardware.

An industrial partner (FLUPOL) was considered for the validation of the object recognition and pose estimation system. This enterprise was interested in the out-coming of the research work, due to their desire of creating a completely autonomous and robotized cell for their production process. Nevertheless, all the object recognition and pose estimation architecture is generic and can be applied to other applications or/and scenarios.

The expected practical implication of this research work is to contribute to the integration of industrial robots in highly dynamic and specialized production lines reducing the company dependency on skilled operators.

Resumo

Nos sistemas de produção modernos o principal desafio é ir de encontro aos pedidos customizados dos clientes, mantendo a qualidade dos produtos e a respetiva taxa de produção. Neste cenário, soluções *state-of-art* que promovem um aumento na flexibilidade e eficiência do sistema produtivo são necessárias. Estas soluções passam por avanços tecnológicos que permitam a redução dos tempos de *set-up* e promovam o aumento da eficiência no desempenho da gama de operações.

Neste cenário, os robôs industriais ainda apresentam algumas limitações que impedem a sua utilização em vastas áreas da indústria. Estas limitações estão diretamente relacionadas com a flexibilidade de programação e adaptação das trajetórias com o meio ambiente e produto alvo de manipulação.

Desta forma, o objetivo da tese de doutoramento aqui apresentada, prende-se com o desenvolvimento de um algoritmo robusto de localização e reconhecimento de geometrias baseado no seu modelo 3D. Estes requisitos emergentes foram levantados pelas necessidades específicas de empresas, as quais procuram a robotização da sua linha de produção, caracterizada por uma vasta gama de pequenas séries de produtos fabricados em simultâneo.

Devido ao seu processo produtivo, muitas destas empresas vêm inviabilizada a solução baseada noutro tipo de identificadores, nomeadamente RFID ou códigos de barras. Estas limitações são impostas pela aplicação de tratamentos de pintura ou de altas temperaturas aos seus produtos. Além disso, a fraca qualidade mecânica dos seus transportadores, impossibilita que seja garantido um pré-posicionamento dos produtos alvo de interação. Desta forma, com o desenvolvimento de um sistema de reconhecimento de geometrias e estimação do posicionamento do objeto alvo de manipulação, permitirá a selecção autónoma por parte do robô do programa a ser executado bem como realizar o respectivo ajuste da trajetória. Para este efeito, um sistema em cascata baseado nos algoritmos *Support Vector Machine* e *Perfect Match* foi desenvolvido.

A implicação prática esperada deste trabalho, é contribuir para a integração de robôs industriais em linhas altamente dinâmicas e especializadas, reduzindo a dependência das empresas em operadores qualificados. Em todo o trabalho de investigação houve a participação como, parte interessada na solução, de uma empresa de revestimento Portuguesa - FLUPOL.

Acknowledges

First and foremost I want to thank my advisers Prof. Dr. António Paulo Gomes Mendes Moreira and Prof. Dr. Vítor Manuel Ferreira dos Santos for the excellent guidance, criticism and support along all the work developed. Their contributions were very important for the success of this Ph.D thesis.

I would like also to express my gratitude to Prof. Dr. António Paulo Gomes Mendes Moreira for the opportunity given so that I could perform my research and studies in ROBIS research group (Robótica e Sistemas Inteligentes), belonging to INESC-TEC.

I would like to thanks my colleagues and members of ROBIS/INESC-TEC, for their immensely contribution to my personal and professional time at this research group. They have been a source of friendships as well as good advise and collaboration. Special thanks for Andry Pinto, Marcos Ferreira and Germano Veiga. To Andry Pinto, for his important collaboration in the beginning of my Ph.D. To Germano Veiga for its criticism and shared experience. My sincere thanks to Marcos Ferreira for his friendship and for the many hours spent at the industrial prototype. His support in the construction and validation of the industrial cell was a very important contribution for the success of the presented research work. To José Bandeira, Pedro Bandeira and all FLUPOL workers for believing and for the availability to cooperate and contribute to the success of this work.

A very special thanks to Joana Mara Marques Monteiro by believing, by the strength and support shown all over the Ph.D. thesis. And finally to my parents for the opportunity, dedication, understanding and encouragement at all times.

I hope that I can return in the future all the support and friendship.

My sincere thanks and gratitude to everyone.

Luís F. Rocha

Official Acknowledgements

The work presented in this thesis, being part of the Project PRODUTECH PTI (number 13851) - New Processes and Innovative Technologies for the Production Technologies Industry, has been partly funded by the Incentive System for Technology Research and Development in Companies (SI I&DT), under the Competitive Factors Thematic Operational Programme, of the Portuguese National Strategic Reference Framework, and EU's European Regional Development Fund. The author also thanks the FCT (Fundação ao para a Ciência e Tecnologia) for supporting this work through the project PTDC/EME-CRO/114595/2009 - High-Level programming for industrial robotic cells: capturing human body motion.

And finally but not least, thanks to project QREN for supporting this work through SIIARI - Sistema para o Incremento da Inteligência Artificial em Robótica Industrial (QREN - Sistema de Incentivos à Investigação e Desenvolvimento Tecnológico, Projectos de I&DT Empresas em Co-promoção 2010-2012).

COMPETE
PROGRAMA OPERACIONAL FACTORES DE COMPETITIVIDADE

QUADRO DE REFERÊNCIA ESTRATÉGICO NACIONAL
PORTUGAL 2007.2013

UNIÃO EUROPEIA
Fundo Europeu de Desenvolvimento Regional

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

INESCTEC
TECHNOLOGY & SCIENCE
ASSOCIATE LABORATORY PORTUGAL

"A problem well stated is a problem half solved"

Charles F. Kettering

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

SME	Small Medium Enterprises
FMS	Flexible Manufacturing System
PM	Perfect Match
ToF	Time-of-Flight
LRF	Laser Range Finder
kNN	k-Nearest Neighbor
NN	Neural Network
SVM	Support Vector Machine
KKT	Karush-Kuhn-Tucker
FT	Fourier Transform
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
RPROP	Resilient Back-Propagation
DOF	Degrees of Freedom
PFH	Point Feature Histogram
FPFH	Fast Point Feature Histogram
VFH	Viewpoint Feature Histogram

Chapter 1

Introduction

1.1 Motivation

Industrial manufacturing has always taken into consideration competitive factors, such as time, cost and quality. However, manufacturing is more and more characterised by customisation, which can be accomplished by reducing lot sizes, increasing variety and specific products manufactured in short periods of time.

Another specificity of next-generation manufacturing is the increasing complexity of products. Instead of rigid infrastructures or inflexible approaches, adaptive manufacturing is envisioned as a new paradigm aiming for continuous and permanent adaptation of all entities inherent to manufacturing systems, namely the ones related to manufacturing resources, materials and information flows. Moreover, the increasing globalisation and as a consequence increasing competition asks for products quality improvement and competitive costs. The understanding of these multidimensional challenges leads to the use of techniques and tools to improve manufacturing processes and to decrease and eliminate non-value activities (such as transportation tasks, machinery set-up times and task scheduling optimizations) - Flexible Manufacturing Systems (FMS) [20, 74].

In a recent past, and due to the constant evolution of technology, sophisticated machinery is increasingly available, which allows manufacturing firms to achieve significant process and set-up times reduction, contributing in this way to the so desired flexible production process [70]. Therefore, and to keep their competitiveness in the market, industrial manipulators must follow this technology evolution or otherwise they will only be used for repetitive processes or mass production strategies [24].

One of their most limiting features accepted as such, from a flexible manufacturing point of view, is their programming procedure. Typically this programming is fairly time consuming process and represents a high investment, unaffordable for SME's. These limitations are imposed by the complexity of their teach pendant that needs to contain all the commands available to interact with the industrial manipulator as well as their programming simulator, which is only accessible to expert personal. To overcome these difficulties, in the last few years robot manufacturers have tried to improve the teach pendant interface, making it more intuitive by implementing an icon-based programming, 6D joystick and coloured touch screens.

However, the programming procedure is not the only obstacle of industrial manipulators that prevent their widespread use in diversified fields of industry. The lack of capacity that they demonstrate in detecting and locating three-dimensional objects, as well as the stiffness of previously defined motion paths, makes it impossible for them to be applied in highly dynamic production environments. These characteristics are at odds with the current state of the industry. Therefore, other approaches are required where the developments verified at the level of sensors and actuators open possibilities for designing and developing new solutions particularly through integration.

Hence, the scientific developments achieved as part of this work focus on the flexibility enhancement of industrial cells, especially robot conveyor assembly lines, where the key elements are industrial manipulators. The idea is to create an object recognition and pose estimation system that complements the lack of perception capabilities presented by industrial robots. The developed system is a robot independent, and will be responsible for sending to the robot the object's Id. and the respective estimated pose. This will allow the correct manipulation program to be uploaded automatically, and the performance of correspondent coordinate system corrections.

1.2 Work Objectives - Summary

Considering the presented research problem, which aims at providing industrial manipulators with the capacity to recognize objects and identify possible position and orientation changes, 3D modelling sensors as well as approaches from pattern recognition are to be examined.

The success of this work will contribute for, not only the reduction of the company dependency on skilled operators, but also in the enhancement of the industrial manipulators flexibility opening possibilities for their use in different fields of industry.

Thus to achieve these objectives, this thesis is organized in the following steps:

- Survey of the state-of-the-art on 3D Model Reconstruction Sensors, Features Extraction and Object recognition topics, making especially attention on industrial robotic applications.

- In the Machine vision topic, select the best approach considering the objects to recognize and how they are displaced in the environment.
- Identification/Development and integration of techniques for object recognition, exploring pattern recognition field (machine learning algorithms, point cloud analysis and model template matching). It is expected that, with this approach it is given to the robot the ability to recognize the part to be manipulated. Hence, allowing the autonomous selections of the right robot trajectory program.
- Estimate the object displacement and adjust automatically the robot trajectory.
- Test and validate the results: laboratory case study and industrial use.

1.3 The Overall Challenge

Industrial robots have in their internal database a set of manipulation programs - trajectories. These trajectories are rigid and specific for each object type. This fact implies that, for each product in hand it is necessary to load a specific program - oriented for mass production strategies. In other words, for the execution of the a particular task the robot needs to "know" the position of the object and its Id.. In Figure 1.1 [1], it is presented an illustration of the discussed problem.

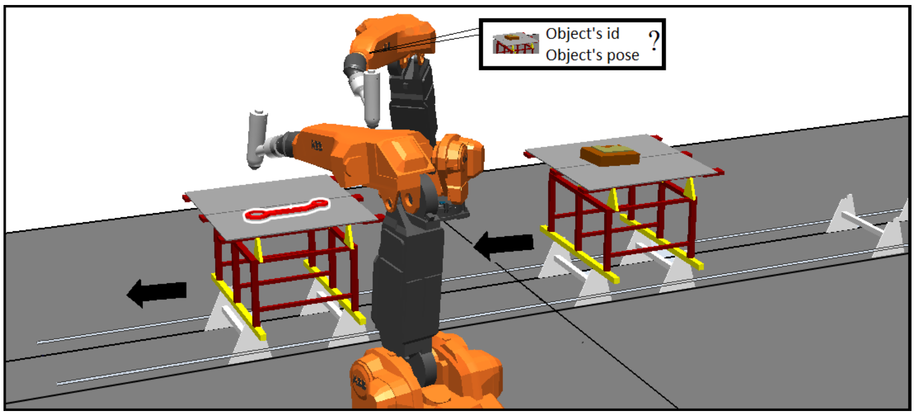


Figure 1.1: Industrial robot challenge illustration - Identification and trajectory program selection and adaptation according to target object id and pose.

To overcome this barrier and to consider highly dynamic production processes, current state-of-art solutions adopted by enterprises involve the use of bar codes, RFID sensors or serial numbers that characterize completely the product in hand. Thus, allowing the robot to automatically load the right program to execute.

[1]Illustration performed using the ABB RobotStudio

However, this kind of solutions are only adequate in production chains fully automated where each machine is connected to a central database. In Figure 1.2, it is presented one of the most well known process characterized by a fully automated production line, where at each time, at each station, it is known which product is actually being processed and which are its specifications.

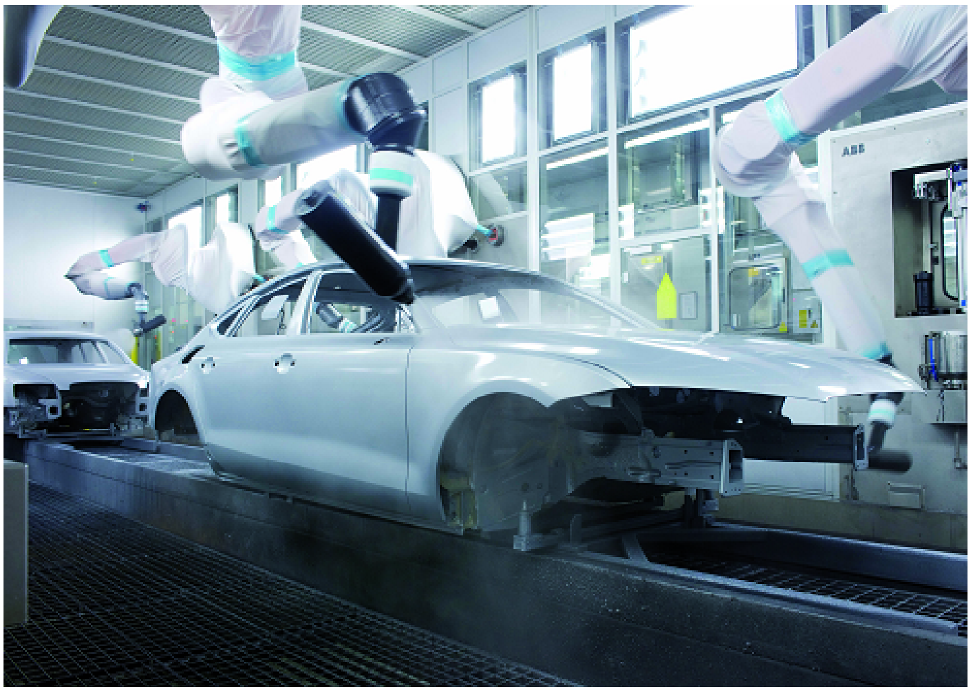


Figure 1.2: Audi's car painting production line.

In these cases, information transition to all machinery present in the shop floor is simple to execute. So if a specific product customization is needed, like coating/colour type, only the respective product database needs to be changed. Or in the case where different product models are produced in the same production line the robot easily have access to the part specifications information.

Another application example is the one present in Figure 1.3, where different parts are processed at the same time in the production line; therefore, the robot has to have access to the part information Id..

Although bar code or RFID are completely valid, reliable and robust solution, for SMEs where completely automated process is not installed due to its high costs or for specific production processes that impossibility's the use of the referred sensors/tags (paint, bath, heat treatment), the problem is not solved calling for the development of different solutions (especially low cost ones).

As referred before, and adding to the problem of object recognition, estimating objects pose is also needed. To overcome this problem, common solutions involve a

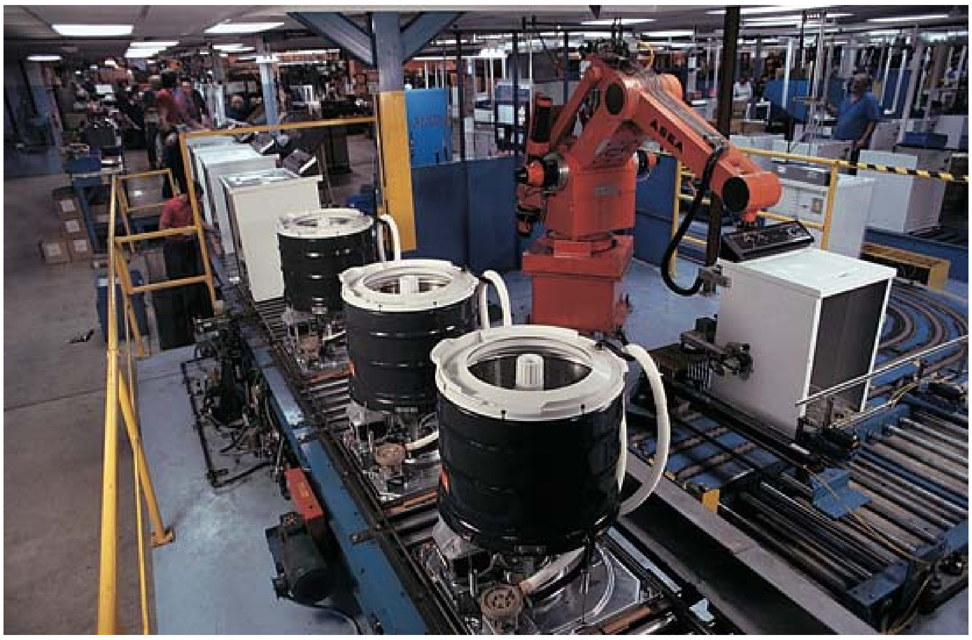


Figure 1.3: Speed Queen company assembly line.

mechanical pre-definition of a pose set for each type of part, as seen in Figures 1.2 and 1.3. However, in many production lines, especially in SME's generally characterized by an unstructured material flow majorly due to mechanical conveyor limitations (type of objects are randomly located), this predefined position set is not so trivial to assure or only can be guarantee if high investments are performed in the production line. See Figure 1.4 for some industrial application examples. Therefore, the need for object's pose estimation (position and rotation) rises.

As it is possible to conclude, other solutions need to be explored and applied to industrial conveyor lines. Even more when short production series are the main core of today's industrial environment.

1.4 Thesis Main Contributions

Presented the target problems, in this thesis it is proposed a vision based solution that contributes to the object (target of robot interaction) identification and pose estimation and that could be used by SME's. The solution needs to be low cost and easily adaptable to different types of objects and applications. Also, high reliability and good precision in objects pose determination and high classification ratios are mandatory needs. The idea is to create a complementary system responsible for acquiring the object's model, classify it and estimate its pose.

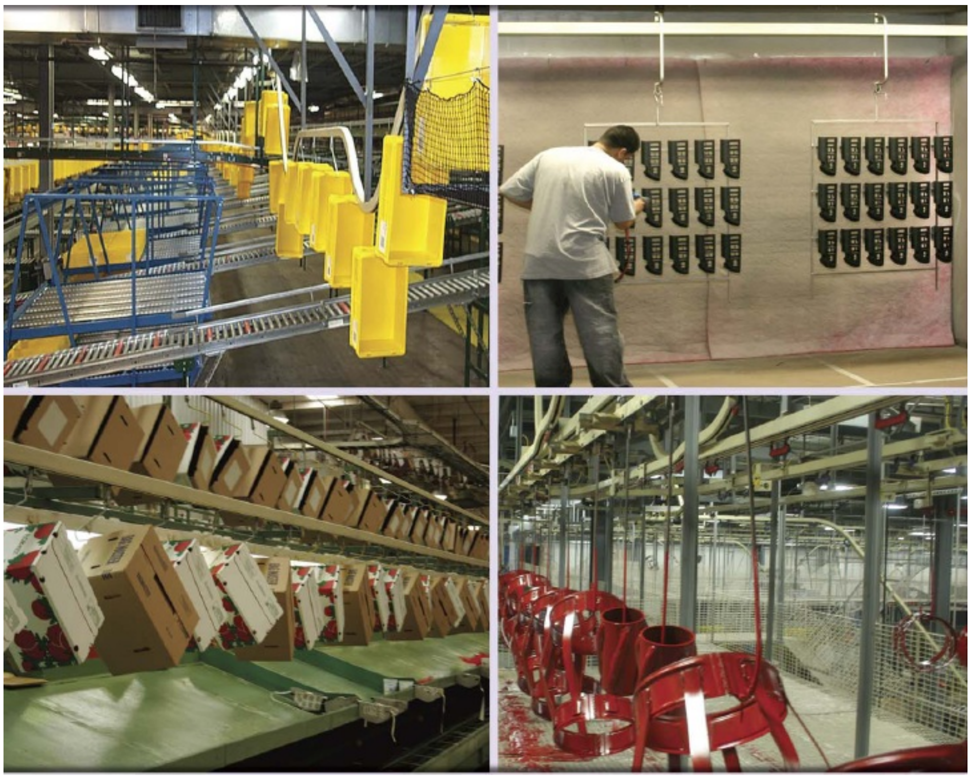


Figure 1.4: Industrial applications examples where it is difficult to ensure product pose - KMI systems inc.

In the end, the industrial robot is informed about the objects id and pose, allowing the correct program loading and pose adjustments for the robot-part interaction task. Note that, the system that is proposed in this thesis does not interfere with the normal robot task execution. If a solution robot dependent (like the eye in hand approach [54]) was considered, the production time could be affected.

Furthermore, this thesis has as main target industrial applications where solutions reliability and robustness are a must, since a minor error or fault in the production system results in high monetary losses for the enterprise. Even more when, with the actual world crisis products selling prices are strangled to a minimum leaving the company with short monetary production flexibility.

In this sense, and from the research work developed, the following scientific contributions were achieved:

- Introduction of an algorithm for object classification purpose and pose estimation based in the object 3D model - The Perfect Match (PM) algorithm. This algorithm is based in the direct geometric match of the object 3D model, and does not use features information. Features are a representation of the models information. However, the extraction of intrinsic and unique information present in the 3D models information, and to guarantee these features' quality can be difficult to achieve. With PM algorithm a robust to outliers, computational efficient and reliable classification and simultaneous relative pose estimation algorithm is achieved. Furthermore, the object pose estimation is performed considering only one template (in contrast with many state-of-the-art solutions).

  Although its degree of freedom limitation (only 3 DoF), imposed by the exponential increase of the computation cost with the computed dimensions, it can be applied in most industrial environments. Another problem with PM algorithm is the number of matchings necessary to perform to classify a specific object, that is proportional with the number of models recorded in the database. Support Vector Machines (SVMs) were introduced into the solution, allowing the creation of a fixed number of models subset where PM is applied.

- Development of a generic architecture that can be applied in many applications or scenarios (like bin peaking or grasping objects at home environment) and is independent of the 3D model sensor used.

As a summary, this work tries to contribute to the flexibility enhancement of industrial robots by adding the perception skill, a weak point identified by many, contributing also for the widespread of industrial robots application areas and strategical scenarios. The work is developed considering an industrial coating problem; however, the modularity of the solution was built so it can be extrapolated to other industrial applications.

Chapter 2

An Architecture for Part Recognition and Pose Estimation

Presented in Chapter 1 the main objective of this thesis, this Chapter starts by presenting a state-of-the-art revision on object recognition and pose estimation streams. Then, all the architecture built for classifier model training including features extraction, the machine learning methods considered, classifier model selection and assessment will also be discussed.

Since the idea is to build an architecture reliable for object recognition and also pose estimation, the Perfect Match algorithm [34] was explored and added to the architecture. This algorithm is used nowadays in robot soccer competition for robot localisation. It uses a previous computed distance map of the soccer field, used has template, and match it with the actual field view of the robot. Therefore, the idea is to extrapolate its application to object recognition and pose estimation.

2.1 Related Work on Object Recognition

Considering a 2D/3D model of the scene foreground, these models contain a significant amount of information that can be analysed making it possible to extract the fundamental characteristics specific of each object that needs to be identified.

Object recognition is coarsely composed of two steps: (1) feature extraction and (2) object classification.

2.1.1 Feature Extraction

Considering the problem presented and already discussed, and for objects shape distinguishing (texture-less objects are common at industrial environment), some well known approaches from image analysis field are available.

Yang Mingqiang *et. al* in [44] presents a survey of shape feature extraction techniques. The authors refer that an efficient shape features must present some essential properties:

- identifiability - objects shape which are found perceptually similar by human have the same feature different from the others;
- translation, rotation and scale invariance - object pose, rotation and scale changes do not influence the features extracted value;
- noise resistance - feature robust to noise;
- occlusion invariance - when some part of the object is hidden, the features of the remaining parts must not change compared with the original object;
- reliable - as long as one dealing with the same pattern, the extracted features must remain the same;
- independence - two features have to be statistically independent;

In the referred article, the authors divided the shape-based feature extraction according to their processing approaches.

Therefore, to discriminate different objects it is simply necessary to distinguish the parameters/features value that belong to each class preserving the image's information [51].

In the image analysis field, and for feature extraction purpose, one of the most used for evaluating object's shape is the determination of the invariant moments since they do not depend on scaling, translation and rotation [17]. Although one of the most well known descriptor others like Fourier, eigenvalues of Dirichlet Laplacian [28], wavelet, primitives extraction (see figure 2.1) have been developed to describe the shape of different objects [50].

2.1.2 Object Classification

Having captured unique features with some of the techniques referred before, object classification techniques must be applied. In this area the most well know strategies are from pattern recognition field, where Machine Learning based methods (like k-Nearest

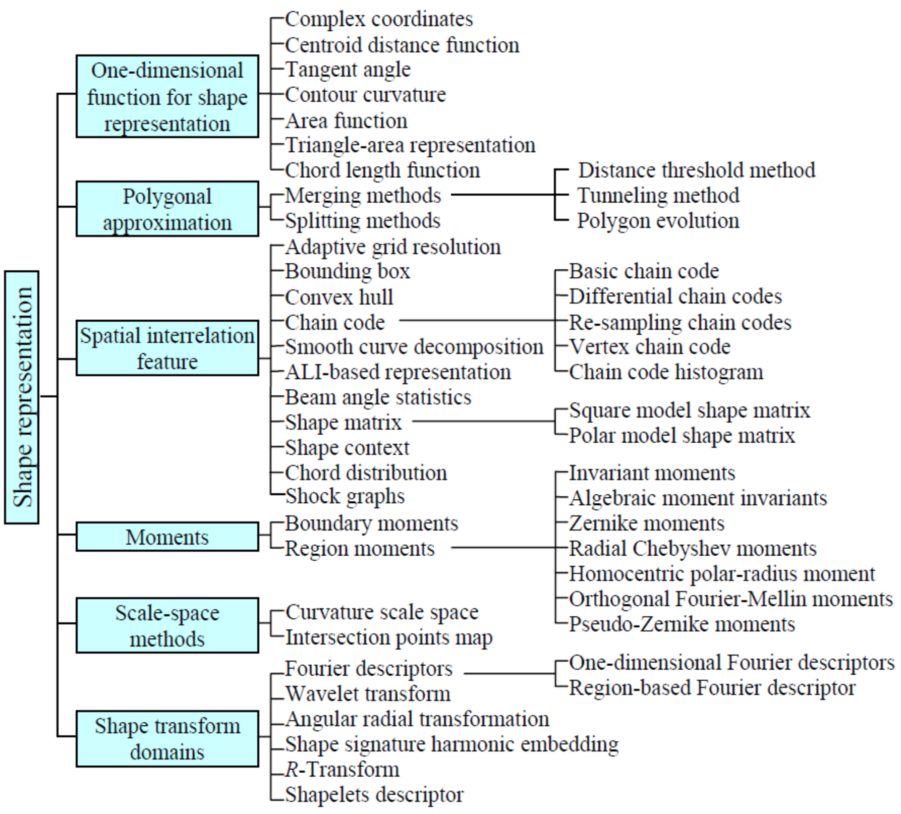


Figure 2.1: An overview of shape description techniques [44].

Neighbour (kNN), Support Vector Machine (SVM), Neural Networks (NN), Hidden Markov Models and Naive Bayes), template matching and more recently Point Cloud are the main focus.

In the context of the presented industrial application, machine learning algorithms based on supervised learning are the best approaches, since the type of objects are clearly defined and have a specific robot program to execute.

Therefore in the next section, and using the referred approaches, interesting state-of-the-art research work will be presented.

2.1.2.1 Feature Extraction plus Machine Learning Algorithms

Starting with the kNN one interesting work, and that evolves object recognition and pose estimation (6DOF), is the one presented in [64]. In this work, the authors propose a new descriptor data that encodes geometry and viewpoint - Viewpoint Feature Histogram (VFH) considering direct Point Cloud analysis. In the training phase their approach consists into extracting different viewpoints of the object. Then, for each sample is computed the correspondent VFH descriptor, which is then recorded in the database. In the on-line phase, it is computed the VFH descriptor for the object in hand, and a comparison between VFH's (the actual with the recorded ones) is performed by applying the kNN algorithm. With this approach object recognition and pose estimation (the authors refer that the pose precision is accurate enough for robot manipulation) is achieved. The authors tested the algorithm for 60 objects and 54000 scenes having a recognition rate of 98.52%. The only constraints of the objects are rigidity and not reflective or transparent (limitations imposed mainly due to the 3D sensor used). The problem of the approach is that the pose estimation precision is directly dependent on the viewpoints samples rotation resolution that are extracted during the training phase.

On what concerns NNs use, in [53] it is presented an invariant object recognition and pose estimation based in a 2D vision system for robot assembly tasks. To perform this recognition, the authors use a NN which receives as input a descriptive vector called CFD&Pose. This vector is constructed based on a canonic shapes database (saved during a training phase). After shape processing a descriptor vector invariant to location, scaling and rotation is achieved. For objects pose estimation, it is admitted that the objects are placed in a plane surface. Using centroid computation, objects height and orientation, it makes possible robot - object interaction. With this procedure, a 100% recognition rate was achieved although only 3 types of assembly objects were considered.

Amitabh Wahi proposes the use of a NN to recognize rotated irregular shaped edge objects. In their work, a 2D colour image of the objects is captured and then converted to grey-scale and finally to a binary image, so objects boundary can be computed. For training purpose, the objects are rotated every 5 degree obtaining 72 images for each object. The features used are the mean and standard deviation. With 5 degrees object rotation during the training phase the proposed method achieved 90.8% of classification.

For its turn, [29] presents an object recognition method for cell manufacturing system. Their approach is based in multi-view point (2D) of the object which are recorded during the training phase. Then, and using a NN, it is classified and estimated the objects orientation that is the target of grasping procedure. In both of these two works, objects orientation is computed from previous images taken from different viewpoints of the object. Even more, the precision of the on-line estimation of object rotation is directly dependent on the number and resolution of object rotations performed during the teaching phase.

Koker *et al.* presents an industrial machine vision for object recognition [33]. This system was developed based in a CCD camera, invariant moments and a NN, where the main focus was the 2D recognition of objects on a conveyor. The position of the objects is estimated using the centroid, which computation is very sensible to models noise and outliers.

The authors in [25], also refer the use of NN for the introduction of object recognition skills to industrial robots. Their focus is in creating a feature vector that integrates robust invariant features by using object contour's and form. Both features are concatenated and are used as the NN input. The accuracy value obtained for recognizing 4 geometric objects was of 100%. Pose estimation was not considered for the study.

Finally for support Vector Machine, [6] presents a comparison between NNs and SVMs in the application of classifying automotive wheels in an industrial environment. The author refers that the main problem in this application is choosing the correct feature set for each wheel so that even the most similar wheel types are correctly classified. Moreover, robustness to noise and to typical dimensional variations observed in manufactured products is also necessary. Different arrangements of their extracted features were used as input in NN and SVM. In the end, SVM outperformed NNs in terms of both classification accuracy and classification speed. The authors considered the SVM better performance as an expected result although warning that exceptions exist, usually when the user has specific knowledge on the application. However, in these cases the difference between the performance of the two approaches is minimal. Also, the authors refer many applications where normally NNs and SVMs are used: bearing fault detection, breast cancer cell detection, drug classification, image retrieval, signature recognition and others.

Considering now mobile robotics outdoor environment, the authors in [41] apply an SVM classifier to identify pedestrians by using laser range, and vision based data. The paper focuses on the SVM model training presenting two possible solutions: a boosting-SVM scheme, where the number of features increases on each training stage. With this, SVM model classification capability and complexity is controlled in every step; The second approach is denominated SRM-SVM. It follows a learning strategy in which, at each stage, it is selected relevant features using the maximum relevancy and minimal redundancy algorithm referred in the paper. This step allows to control the complexity of the model and to minimize at the same time the risk of misclassification.

Considering the same research scenario Cristiano Premebida et al., in [58], developed a classification algorithm based on a cascade of Machine Learning algorithms. In this architecture algorithms such as SVM-RBF (Radial Basis Function), NN and Naive Bayes were incorporated, hoping with that to reduce the misclassification of pedestrians in a real life scenario. The authors presented also a novel feature propagation scheme, where features are propagated along the cascade classifiers. High classification rates were achieved, 94.1 %, though the authors consider that still exists some space for results improvement. Although with different purposes and scenarios then the one target of this thesis, both works present a valid architecture for SVM training and testing for object recognition purposes.

For his turn, Massimiliano Pontil et al. in [57] assessed the potential of using SVM's as the classifier for 3D object from a single view. The authors concluded that SVM is effectively trained even if the number of examples is much lower than the dimensionality of the object space. The good results reported indicate that SVMs are likely to be very useful for direct (images without feature extraction) 3D object recognition.

Finally, Hao Zhang in [83] uses kNN followed by SVM to perform image recognition. In their approach, a feature vector based in texture and shape is extracted and compared with the training data trough kNN. If the final labelling is in accordance with all of the neighbours the label is accepted if not, it is computed the pairwise distances between the considered neighbours. This distances are converted into a kernel matrix and then used as input in a DAGSVM.

2.1.2.2 Pattern Recognition and Template Matching

Although Machine Learning is a valid approach, others from pattern recognition field like template matching are often used.

For the recognition of the light signal for the Autonomous Driving Competition Robotica 2011, [82] presents a combination of two techniques based on blob analysis and pattern analysis. Their approach consists in applying blob analysis to extract the properties of a pre-segmented image regions. Then and to perform an adequate detection of signs, a comparison with some reference symbols was performed. Therefore, a pattern of properties for the symbols (regions) was defined, and the Mahalanobis distance was computed for each symbol candidate. The authors refer an achievement of almost 100% for the recognition accuracy of symbols for distances to the object up to 2m. Although these good results, the accuracy was dependent on surrounding light condition due to the use of CCD Cameras.

Manuel Wopfner *et al.*, in [80] presents a recognition approach based again in the squared Mahalanobis distance for each object to the object class recorded. The features used in this approach are the dimensions of the object bounding box. However, the objects are relatively well separable. Note that, the work presented refers directly to the recognition of objects for direct manipulation by a 5 axis robot in a human environment. Although, focusing in a home scenario and not industrial one.

M. Oliveira and V. Santos in [48] and [49] presents an interesting work which deals with the problem of tracking fully dynamic objects (ex. car detection) based on Harr features, that are used as a single view identifier and complemented by template matching to track the previously classified object.

In [47], the authors focus in the bin picking problem. The main difficulties in this application are: objects recognition (if a scene with different types of object is considered) and pose estimation, occlusion of objects parts, grasp selection and motion planning. For object recognition, their approach is based on sub-graph matching. In other words, the authors convert the searched object and the scanned scene into an annotated graph. Each node of the graph is instantiated with the detected 3D shape primitives and respective 2D contours. Edges connect primitives which are neighboured in space and also store their relative position. Therefore, objects in the scene are located by comparing the graph of the object with the graph of the scene. The objects pose is computed by the partial matches between the model and scene graph. For each primitive match, one or more of the 6 DOF are computed. Although good results achieved the cognition phase times presented are a problem for industrial applications. Also, object recognition in a scene with different types was not considered.

Michele Fanzi et al. in [14] presents a 3D object recognition and pose estimation algorithm for multiple objects. Their approach consists in computing SIFT features from a set of training images covering the objects (multi-view points). In the on-line phase, the model recorded during the training phase is matched in the cluster (off the scene) using SIFT matching and the RANSAC method proposed by the authors. For very similar object, SIFT may find numerous matchings for interesting points. Therefore, this approach, and in most of the applications, have always to be complemented by the computation of other features, being also more suitable for objects with well defined texture.

Another approach interesting for consideration, is to use the models Point cloud directly (retrieved by the 2.5D or 3D sensors) and use the matching algorithm that some middle size robotic teams use for locating accurately the robot in the soccer field. The approach is based on an efficient numerical approach to find the locally best match between features/measures obtained in the camera image and a map of the field. For this purpose, they apply the resilient propagation algorithm to minimize the matching error [34]. The matching in this algorithm is made at the 2D level; therefore, extrapolation for 3D is needed. Note that, although a vast number of research works in the recognition area, only a few refers directly to industrial direct application.

2.2 Features Considered for the Proposed Architecture

Considering the state-of-the-art review presented, some of the most used machine learning algorithms will be applied and integrated in the proposed system architecture, making it possible to distinguish the objects of the different classes (multi classification problem). Therefore, these methods require the extraction of object features that will be used as an input vector. The quality of these extracted features, or in other words, how much better these features characterize well each object type, better will be the generalization error (classification accuracy) of the trained model.

With this objective, and from the state-of-the-art, it was selected the following main features: Hu moments and Fourier Transform. Besides those, some additional features were also added: width, height, energy, entropy and eccentricity of the object models.

2.2.1 Hu Moments

As the first approach for feature extraction, invariant moments were considered mainly due to their immunity to object rotation, scale and translation.

- Ordinary Moments

  Considering a model as two-dimensional density distribution, $f_{XY}(x,y)$, continuous real function and a finite non-zero integral, the general moments, M_{pq}, can be viewed as "projections" of a function onto a polynomial basis [51]:

$$Mpq = \int\int_D p_{pq} f_{XY}(x,y) dxdy \tag{2.1}$$

  Where $p_{pq}(x,y)$ is the polynomial basis defined in the domain, D, and $p+q$ is called the moment order (p and q are non-negative integers). From Hu's unique theorem, there are moments of all orders and the sequences of M_{pq} are uniquely determined by $f_{XY}(x,y)$. Reciprocally, those moments define uniquely the function $f_{XY}(x,y)$. In other words, the moments have the property of preserving the image's information [51].

- Geometric Moments

  Using $p_{pq}(x,y) = x^p y^q$ as the basis function the geometric moment, m_{pq}, will be defined by (considering the discrete form):

$$m_{pq} = \sum_{i=1}^{M} \sum_{j=1}^{N} x_i^p y_j^q f_{XY}(x_i, y_j) \tag{2.2}$$

  Geometric moments of low orders are widely used to determine intuitive parameters, for instance the zero order moment that represents the "*mass*" of the image or area in binary images, m_{00}, and the first order moments (m_{10} and m_{01}) that provide information on the "*centroid*" location, $\bar{x}$ and $\bar{y}$:

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad , \quad \bar{y} = \frac{m_{01}}{m_{00}}. \tag{2.3}$$

- Central geometric moments

  If the object's centroid coincides with the origin of the coordinate system, the moments will be invariant to the translation and rotation [17]. That can be achieved by calculating the central geometric moments:

$$\mu_{pq} = \sum_{i=1}^{M} \sum_{j=1}^{N} (x_i - \bar{x})^p (y_j - \bar{y})^q f_{XY}(x_i, y_j) \tag{2.4}$$

  By computing the second order moments, it is possible to determine the covariance matrix, J, and the rotation angle[1], ϕ:

$$J = \begin{bmatrix} \frac{\mu_{20}}{m_{00}} & \frac{\mu_{11}}{m_{00}} \\ \frac{\mu_{11}}{m_{00}} & \frac{\mu_{02}}{m_{00}} \end{bmatrix} \tag{2.5}$$

$$\phi = \frac{1}{2} arctan \left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right) \tag{2.6}$$

  Eccentricity can also be calculated with the eigenvalues (λ_1, λ_2), of the covariance matrix: *Eccentricity*=$\sqrt{1 - \frac{\lambda_2}{\lambda_1}}$. This value represents a relation between the semi-major and semi-minor axes length of the image.

- Normalized Central geometric moments

  To make the moments invariant to translation, rotation and scaling, it is necessary to perform a proper normalization of the central geometric moments. The formula used more often for this normalization is [17]:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\left(\frac{p+q}{2}\right)+1}} \tag{2.7}$$

  The result, η_{pq} is called *normalized central geometric moments*, and is normally used to determine the rotation invariant moments, also known as Hu's invariant moments.

[1] Angle between the x-axis and the axes associated to the higher eigenvector of J.

- Hu's invariant moments

  With these moments, invariance under translation, changes in scale, and also rotation of the object is achieved [40].

$$
\begin{aligned}
&I_1 = \eta_{20} + \eta_{02}\\
&I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2\\
&I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2\\
&I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2\\
&I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\left((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right)\\
&\quad +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right)\\
&I_6 = (\eta_{20} - \eta_{02})\left((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right) + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\\
&I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})\left((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right)\\
&\quad -(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})\left(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right)
\end{aligned}
$$

  There is also an extended eight invariant moment defined as:

$$I8 = \eta_{11}\left((\eta_{30} + \eta_{12})^2 - (\eta_{03} + \eta_{21})^2\right) - (\eta_{20} - \eta_{02})(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21})$$

2.2.2 Fourier Transform

The Fourier Transform (FT) was also selected so the horizontal and vertical shape pattern of the object to recognize could be analysed (these objects will be presented chapters ahead in this thesis).

Returning to Fourier Transform, FT decomposes a signal into a representation in the frequency domain. Its mathematical formulation is defined by the following equations:

$$x(t) = \int_{-\infty}^{+\infty} X(f)e^{j2\pi ft}df, \tag{2.8}$$

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}dt, \tag{2.9}$$

Where t and f stand for time and frequency respectively, and x and X denote the signal at hand in the time domain and frequency domain respectively, defining in this way the so called Fourier transform of $x(t)$ and the Inverse Fourier transform of $X(f)$.

The Discrete Fourier Transform (DFT) is defined by:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j(2\pi/N)kn} \qquad (k = 0, 1, 2, ..., N-1) \tag{2.10}$$

Whose inverse transform is computed from:

$$x[n] = \frac{1}{N}\sum_{n=0}^{N-1} X[k]e^{j(2\pi/N)kn} \qquad (k = 0, 1, 2, ..., N-1) \tag{2.11}$$

Although the original signal sequence in time-domain consists of real numbers, the DFT of a signal is a sequence of complex numbers. Therefore, to make possible the use of DFT as a feature vector for a machine learning model, the most common way is to multiply the DFT sequence by its complex conjugate. Therefore, the multiplied sequence becomes a sequence of real numbers (the power of the magnitudes of the original sequence). Moreover, the last half of the sequence is a duplication of the first half.

There exists a problem with the DFT approach: Leakage. When a DFT signal is computed, it is assumed that it is periodic in the data block. If it is not the case, then the resulting frequency spectrum suffers from leakage. Leakage causes signal energy to scatter out over a wide frequency range in the DFT when it should be in a narrow frequency range.

2.2.2.1 Window Function

To overcome the non-periodic signal problem, a window function is normally applied to the signal.

A window function is a function shaped so that it is exactly zero at the beginning and at the end, and has a certain shape in between the data block. In this sense, the effects of spectral leakage can be minimised by reducing the discontinuities at the end point of the signal. This leads to the idea of multiplying the sampled signal by a function that smoothly reduces the signal to zero at the end points, thus avoiding discontinuities. The process of multiplying the signal data by a function that smoothly approaches zero at both ends is called "windowing", while the multiplying function is called a "window" function.

2.2.2.2 Fast Fourier Transform (FFT)

For DFT computation Fast Fourier Transform (FFT) was used. FFT is a much more efficient algorithm that requires a low computational effort to execute the Discrete Fourier Transform (parallel processing) [61]. However, it presents a constraint as the number of samples of the input waveform need to be an integer power of two (2^N).

As the reader may already understood, in the procedure here presented the signal time variable will be replaced by space (point cloud 3D data). In other words, the FFT will be used to verify if the part presents any horizontal or vertical geometrical pattern. To perform this analysis horizontal and vertical cuts in the 3D Model were made.

2.2.3 Other Considered Features

Besides the Hu moments and Fourier components, the other considered features are:

- Models Dimensions - height and width extracted from the 3D model.
- Energy and Entropy of the 3D model (converted to a $2D$ grey scale image) histogram (h) [19]. The energy of an image histogram represents the image texture uniformity, i.e., reflects the repeatability of some texture in the objects image. It can be computed by the following equation:

  $$ene = \sum_{i=0}^{255} h(i)^2 \tag{2.12}$$

  where i denotes grey-scale value represented by an 8 bits integer.

  The entropy represents the uncertainty in the image values (histogram dispersion). This value is equal to zero when all image pixels have the same intensity, by contrast will be maximum when an image contains the same quantity of pixels for all intensities.

  It can be computed by the following equation:

  $$ent = -\sum_{i=0}^{255} (h(i) \cdot log_2(h(i))) \tag{2.13}$$

2.3 Feature Selection

Having presented the features that are to be extracted from the objects models, in this section it is introduced the feature selection concepts.

Feature selection is a process often used in data mining applications. A subset of features is selected from all the available data according to a certain criteria. By reducing the number of features and aspects such as redundancy, irrelevant, or noisy data, it is possible to speed up the learning algorithms and improve their performance [84]. The best subset contains the minimum number of features that contribute to increase the accuracy value. When there are many irrelevant features, the learning models tend to over-fit the data and to be less comprehensible. This is an important preprocessing stage where the final goal is to reduce the dimension of the data by finding a small set of important features that offers good classification performance and reduces the complexity of the classification models.

There are two main approaches to reduce the number of features: feature selection and feature transformation [43]. Feature transformation methods turn data from the original high-dimensional feature space into a new space with reduced dimensionality. In contrast, feature selection algorithms select a subset of features from the original feature set, keeping their original meaning. The selection criteria usually involve the minimization of a specific measure (predictive error) for models to fit different subsets. The algorithms search for a subset of features that optimize the predictive models, subject to some constraints such as required/excluded features and subset size.

Feature selection can be organized into three main families: filter, wrapper and embedded methods [84, 43].

- Filter: rely on general characteristics of the data to evaluate and select the feature subsets without involving any learning algorithm (features are evaluated only through the intrinsic properties of the data).
- Wrapper: search for the features that better fit for the chosen learning algorithm (evaluate the feature subsets based on the performance of the classifier). It can be significantly slower than filter methods if the learning algorithm takes a long time to run. Usually these are more accurate than filter methods on a particular classifier, however the selected features may not be the most appropriate for other classifiers (poor feature vector generalization).
- Embedded Methods: incorporate feature search and the learning algorithm into a single optimization problem formulation. The selection is performed as a part of the training process, and feature relevance is obtained analytically from the objective.

2.3.1 Wrapper Methods

The selection of a good subset of features could be performed in three ways: exhaustive search (2^D possible combinations, where D is the total number of features), random search methods and heuristic search strategies. In an exhaustive search, all combinations of the feature subset are tested. Although it is possible to find the best or optimal feature subset, it is expensive and in some cases computationally impractical, due to a larger size of the search space.

Random search methods, add some randomness in the searching process to help the escape from local optima, although the reach for the best feature subset is not guaranteed.

Feature selection is an NP-hard optimization problem and for that kind of problems heuristic methods (where a certain guideline is used for the selection) are commonly used. There is one problem for the heuristic approach, there may exist a high order combination of relevant feature subset and for that the optimal solution is also not guaranteed.

There are several heuristic methods (forward selection, backward selection, combined, etc), but for this research work the Simulated Annealing was considered as the selection algorithm, where a worse neighbourhood solution can be momentarily accepted depending on some conditions. This method was the selected one mainly due to the fact that it tries to escape local minimum/maximum, its simplicity of implementation and integration in SVM algorithm and robustness demonstrated in previous works [54]. Although the advantages referred, parameter tuning and the definition of the initial temperature are sometimes difficult to perform.

One quick way to decide which and the number of needed features is to define an Evaluation Criteria - normally the Classifier Error Rate (the number of misclassified observations divided by the number of observations).

Simulated annealing was introduced in combinatorial optimization by Kirkpatrick et al. (1983) [30] and independently by Cerny (1985) [10]. In their articles, the concepts presented are heavily inspired by an analogy between the physical annealing process and the problem of solving large optimization problems.

In condensed matter physics, the annealing process consists of two different steps: first increase the temperature of the hot bath to a maximum value at which the solid melts; then, and as a second step, decrease carefully the temperature of the hot bath until the particles arrange themselves in the ground state of the solid.

During the liquid phase, all the particles rearrange themselves randomly, whereas, in the ground state of the solid, the particles are arranged in a highly structured lattice for which the corresponding energy is minimum. This grounded state is only achieved if the maximum value of the temperature is sufficiently high and the cooling procedure done sufficiently slowly.

Metropolis et al. (1953) introduced a simple algorithm based on Monte Carlo techniques for the simulation of the evolution of solid in a heat bath to achieve a thermal equilibrium. Given the current state i of the solid with energy E_i, then a subsequent state j is generated by applying a perturbation mechanism which transforms the current state into the next state by a small distortion, for instance by the displacement of a particle where the new state energy in defined as E_j. If the difference between $E_j - E_i$ is less than or equal to zero, the state j is accepted as the current state. If the energy difference is greater than zero, then the state j is accepted with a probability given by:

$$p = exp(\frac{E_i - Ej}{k_b * T}) \tag{2.14}$$

Where T denotes the temperature of the heat bath and k_b is a physical constant called the Boltzmann constant.

Considering a combinatorial optimization problem, this algorithm can be used to generate a set of solutions. For that the following equivalences must be made:

- solutions in the combinatorial problem are equivalent to states of the physical system;
- The cost of a solution is equivalent to the energy state;
- A control parameter which plays the role of the temperature is also introduced .

Therefore, it is possible now to leave the physical area and focus only in the formulation of the simulated annealing in terms of local search algorithm.

```
select ($x_k$ , Initial Solution)
select ($T_k$ ,$L_k$)
x* := $x_k$
C* := C($x_k$)
repeat
    for $i := 1$ to $L_k$
        generate a new solution $\boldsymbol{x} \in \boldsymbol{V}(\boldsymbol{x_k})$
        calculate $\Delta C = C(x) - C(x_k)$
        if $\Delta C \leq 0$ then
            $x_k := x$
            if $C(x) < C^*$ then
                $x *:= x$
                $C *:= C(x)$
        else
            if $rand$ $[0,1] < e^{\frac{-\Delta C}{T_k}}$ then
                $\boldsymbol{x_k} := x$
    Update ($L_k$)
    Update ($T_k$)
    Until Reach Stop Criterion
```

Figure 2.2: Simulated Annealing pseudo code.

To simplify the presentation, the optimization problem is assumed to be a minimizing one. In Figure 2.2 is presented the pseudo code of the simulated annealing heuristic, where x presents the state or the combinatorial solution, C the cost of the solution or the state energy, T the temperature inherent to the simulated annealing theory and finally L represents the iteration number.

In the end, it is possible to divide simulated annealing in four steps:

- Initial Solution: Generated using an heuristic or chosen randomly.
- Neighbourhood: Generated randomly or by mutating the current solution.
- Acceptance (considering a minimization problem): Neighbour has a lower cost value, or for neighbours that have higher cost value it is accepted with probability p.
- Stopping Criteria: Maximum CPU time; Solution with a lower value than a threshold; Maximum number of Iterations without improvement; Maximum number if iterations.

The main idea behind this approach is, when optimizing a very large system (i.e. a system with many degrees of freedom), instead of "always" going downhill (minimization problem) or uphill (maximization problem), try to go downhill or uphill "most of the times" (Kirkpatrick, 1983 [30]).

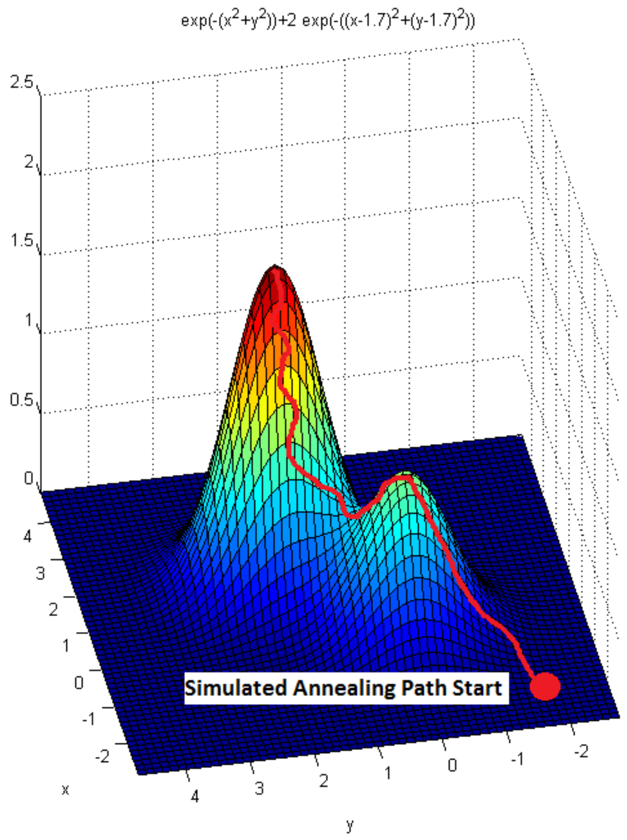


Figure 2.3: Example of the application of the Simulated Annealing heuristic to find the function $exp(-(x^2+y^2))+2exp(-((x-1.7)^2+(y-1.7)^2))$ max value.

In Figure 2.3, it is presented an example of the application of the simulated annealing process in trying to find the function global maximum.

2.4 Machine Learning Approaches

Until now, it was presented the features that are to be extracted from the objects and the feature selection procedure that measures the classification results to rearrange the features combination. Therefore, in this section the classification algorithms considered from the machine learning field will be presented.

The three different Machine Learning approaches considered were: k-Nearest Neighbour, Neural Networks and Support Vector Machine (supervised learning [2]). These algorithms were the selected ones due to their recurrent use for classification purpose in the state-of-the-art, as presented in the beginning of this Chapter.

2.4.1 k-Nearest Neighbour

In the kNN approach, by knowing the k closest neighbours and the classes that they belong to, a distance function is used to select the class that is going to be associate to each test sample. This algorithm is one of the simplest for classification purposes.

Figure 2.4 shows an example of this algorithm application. For this example, it was considered a 2D feature vector (x). In the left side image, it is presented a sample of the population.

Using the kNN, with k equal to 5, and by using the median of the Euclidean distances classes boundaries were computed.

In this algorithm, and for neighbour computation, the mode function is normally used to eliminate noise that could be found in the class features. The median function is used when the classes are well defined/separable. The drawback in this algorithm is that, as k increases and approaches the n value (size of the data), the performance of the classifier will become a statistical baseline and all unknown data will belong to the class most frequently represented in the training set. On the contrary, if the k value is too low (for example $1 <= k <= 3$) the algorithm becomes very sensible to noise. This is not true only if the dataset is completely separable, and the margin between each class is large. If k equal to one is considered, Voronoi tessellation of the training space is achieved [3].

The advantages of this algorithm are the fact that no assumption about the distribution is required ahead of time. However, the number of required samples may be very large

[2]Supervised - construct the model from labelled training data. Unsupervised - refers to the problem of finding the intrinsic structure in unlabelled data.

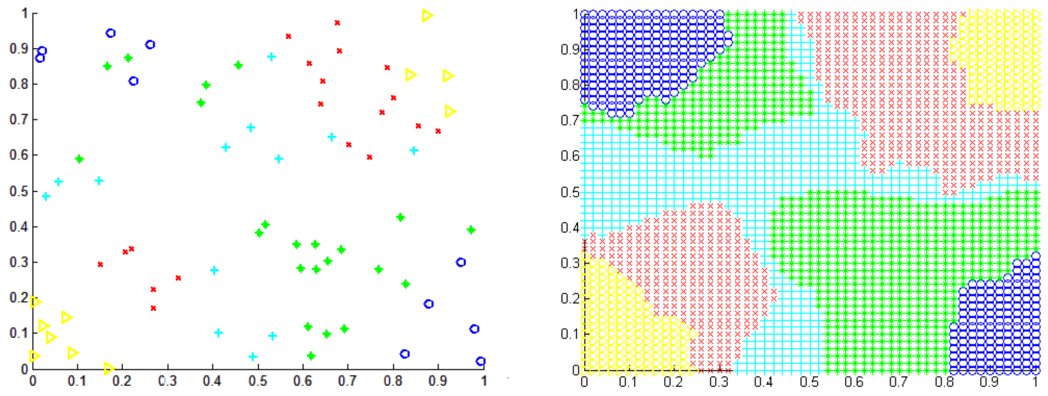


Figure 2.4: Example of the application of K-NN.

(much larger than would be required if the form of the unknown class density distribution were known and could be integrated in the algorithm).

2.4.2 Neural Network

NN are inspired in the human brains composition. NN models have been created by both scientists and engineers. Scientists would like to understand how the brain works, while engineers imitate the brain's structure with the goal of duplicating its function [73]. In a simplistic point of view, NN are an interconnected group of artificial neurons that use mathematical models for information processing, resulting in a network of neurons. The neurons are a simplistic representation that emulates the signal integration and threshold firing behaviour of biological neurons by means of mathematical equations. Like their biological counterpart, artificial neurons are bound together by connections that determine the flow of information between peer neurons. In fact, one of their most powerful features is the ability to generalize from a set of training data, by adapting the weight of the connections between neurons, so the final output activations are correct.

NN have a large range of applications like regression, classification, clustering and optimization.

2.4.2.1 The Artificial Neuron

The artificial neuron is the most basic structure of the NN. Its general mathematics definition can be defined as:

$$y(x) = g(\sum_{i=1}^{n} w_i \chi_i) \tag{2.15}$$

Where x represents an input data with n dimension $(\chi_1, \chi_2.., \chi_n)$, w_i are the weights of each net connection, g the activation function and y the output of the neuron.

If the neuron, or perceptron, was a full replica of the human brain, their output should be binary (0 or 1). However, this is not the usual way they are implemented. For different reasons soft decision function is considered instead of a hard one. In general the activation function return values between 0 and 1 or -1 and 1 (except for the identity function that have no boundaries - normally used for regression). For the inputs and weights there are no boundary limitations. The most commonly used activation functions are the threshold, sigmoid (equation 2.16) and hyperbolic tangent (equation 2.17).

$$g(x) = \frac{1}{1+e^{-x}} \tag{2.16}$$

$$g(x) = tanh(x) \tag{2.17}$$

In summary, the neuron can be illustrated as in Figure 2.5.

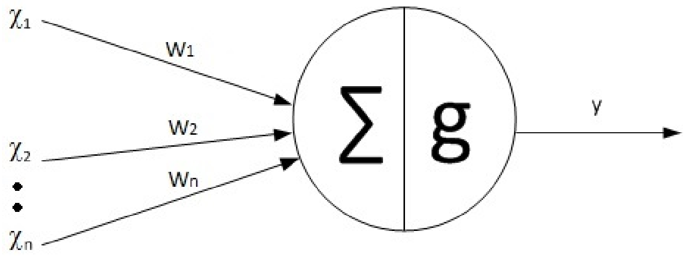


Figure 2.5: Artificial neuron.

2.4.2.2 Multi-layer NN

Multi-layer NN is a network where the neurons (presented in the last section) are ordered in layers (input layer, n hidden layers and the output layer). The authors in [21] describe several kinds of NN, like the recurrent one normally used for time-dependent modulation. In Figure 2.6, it is presented a simple example on a feed-forward fully connected NN. In a feed-forward network the connections only go forward from one layer to the next.

The network presented have 3 layers: an input layer, a hidden layer and an output layer. When constructing the NN, there does not exist any procedure to define the number of neurons in a layer nor to define the number of hidden layers. Therefore, for the construction of a NN experience in their use is a surplus.

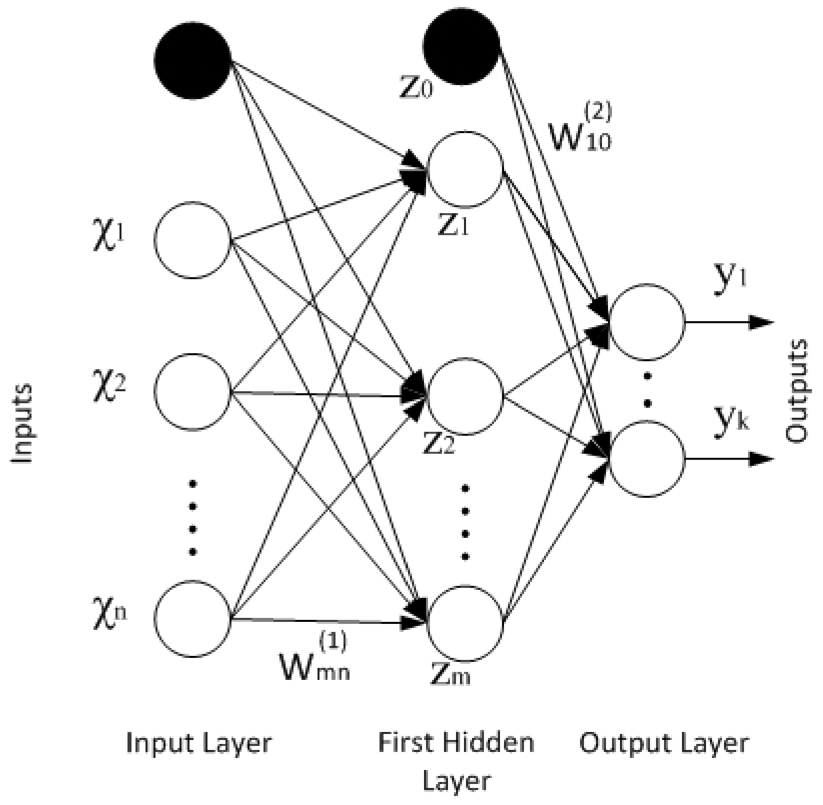


Figure 2.6: Architecture graph of a multi-layer perceptron fully connected with one hidden layers NN.

To make clear the overall functioning of an NN, in the following equations are presented the data-flow for the network presented in Figure 2.6.

Therefore, first is constructed a m linear combination of the inputs χ_1, χ_2, χ_n in the form:

$$a_j = \sum_{i=1}^{n} w_{ji}\chi_i + w_{j0}^{(1)} \tag{2.18}$$

Where $j = 1, ..., m$, and the superscript (1) indicates that the corresponding parameters are in the first "layer" of the network. The parameters $w_{ji}^{(1)}$ are defined as weights and the parameters $w_{j0}^{(1)}$ as biases. The quantities a_j are known as activations.

Next the output of each neuron on the hidden layer is computed, by a non-linear activation function (g(.)) to give:

$$z_j = g(a_j) \tag{2.19}$$

Where z_j refers to the output of each neuron in the hidden layer. Then, these values are again linearly combined to give output unit activations:

$$a_k = \sum_{j=1}^{m} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \tag{2.20}$$

Finally, the output of the network is defined as:

$$y_k = g(a_k) \tag{2.21}$$

2.4.2.3 Training the Neural Network

When training the NN the objective is to adjust the net weights (w_i), enabling the network to give the same outputs as seen in the training data. Although this is true, data over-fitting [3] needs to be avoided. Therefore, the Gradient Descent scheme with Error Back-propagation is normally applied to estimate the net weights [8].

Furthermore, to achieve a good training performance, some issues need to be considered when initializing the NN:

- Initialize the weights with random values near zero. Starting with large weight values often leads to poor solutions. Starting values exactly zero leads to zero derivatives and the algorithm never moves.
- Scaling the input: Standardize all inputs to have mean zero and standard deviation one.
- Generally speaking it is better to have too many hidden units than too few: with too few the model might not have enough flexibility to capture the non-linearities in the data; with too many hidden units, the extra weights can be shrunk toward zero if appropriate regularization is used [22].
- Use cross-validation (explained ahead in this Chapter) to choose the regularization parameter (parameter that makes a trade-off between model complexity and the error of the network).
- Choice of the number of hidden layers is guided by background knowledge and experimentation.

The major disadvantages of the NNs are the need of large training data, long processing time and the definition of the network architecture.

Their major advantages are:

[3] give precise results for the training data, but incorrect results for all other data - poor generalization

- Massive parallelism making them very efficient.
- They can learn and generalize from training data.
- They are very noise tolerant - so they can cope with situations where normal systems would have difficulty.

NN have many fields of application like regression/approximation, clustering, optimization (can be formulated as a neural dynamic process whose behaviour is determined by learning rules and whose stable states provide solutions to the problem) and classification.

2.4.3 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that is capable of inferring the output label of new input values throughout a function (modelled by a set of input examples with a known label - "supervised"). The output of the algorithm is a mathematical function that takes two different values at all points space - binary classifier. The simplest class decision function of SVM is the linear ones. To illustrate this linear classifier, Figure 2.7 present three feasible but different linear decision functions (boundary), that completely separates the two regions of space.

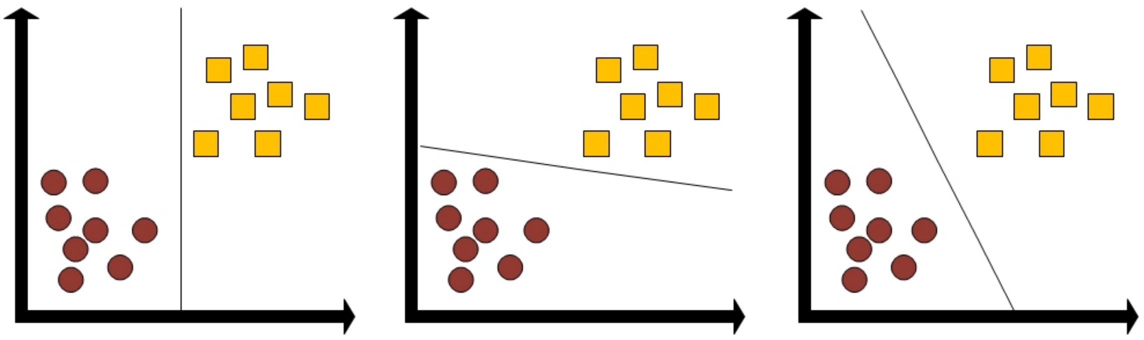


Figure 2.7: A simple 2D classification task.

Although all the presented solutions are feasible, the choice for the best generalisation model (the one that will separate accurately the two sets) needs to be performed.

From a mathematical point of view the decision function can be modelled as:

$$g(x) = sign(f(x)) \tag{2.22}$$

Where,

$$f(x) = < w, x > +b \tag{2.23}$$

With $w = [w_1, w_2, .., w_n]$, $x = [\chi_1, \chi_2, .., \chi_n]^T$, and

$$< w, x > = \sum_{k=1}^{d} w_k \cdot \chi_k \tag{2.24}$$

Being x the input vector, the label $(g(x))$ be either +1 or -1 (binary classifier) and d the dimensionality of the input data.

Therefore given a training set $x_1, x_2 ... x_N$ with the corresponding label $y_1, y_2 ... y_N$ that takes one of the two values (+1,-1), the objective is to choose the parameter w and b of the linear decision functions that generalizes well to unseen examples.

However, this formulation is unsatisfactory as most of the people agree that in Figure 2.7 the superior decision function is the right most one, since it is more robust for measurement noise data. Therefore, the problem formulation needs to be refined. The most interesting decision function is the one that not only correctly separates two classes in the training set, but lies as far from the training examples as possible. In this way, SVM classifiers choose the hyperplane that separates the two classes with a maximum margin (defined as the distance from the hyperplane to the nearest training example).

In order to find this maximum margin, the problem can be formulated as an optimization one. This formulation consists in the maximization or minimization of an objective function with respect to constraints, that need to be fulfilled, while finding the best value for the objective function.

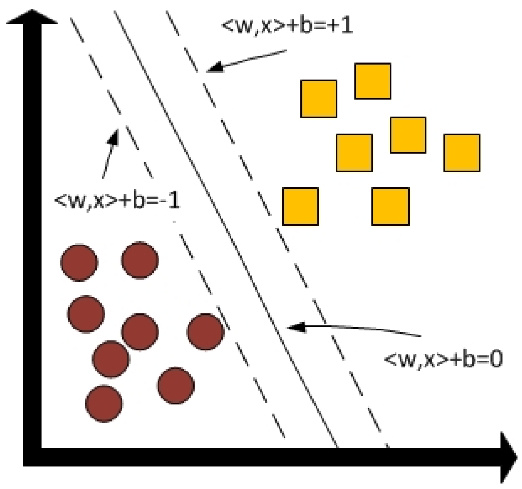


Figure 2.8: Linear separable classification problem.

As stated before, the decision function is parametrized by the vector w and the scalar b. Therefore, to assure that the hyperplane separates correctly the two classes, the following constraints need to be satisfied:

$$< w, x_i > +b > 0, \quad for \quad all \quad y_i = 1 \tag{2.25}$$
$$< w, x_i > +b < 0, \quad for \quad all \quad y_i = -1 \tag{2.26}$$

Or simply,

$$y_i(< w, x_i > +b) > 0, \quad i = 1..N \tag{2.27}$$

However, meeting these constraints is not enough to separate the data optimally - maximize the margin must be ensured. Looking to the Figure 2.8, it is possible to see the decision surface plotted as a solid line.

The constraints referred until now are the same as saying that the data must lie incorrect side of the decision surface (according to the data class label). Note also that, it were plotted two dotted lines which correspond to the hyperplanes where the function $< w, x > +b$ is equal to -1 and +1. Therefore to find the maximum margin hyperplane these dotted lines should be kept parallel and equidistant to the decision surface, maximizing at the same time their distance from one another, while satisfying the constraints that data lie in the correct side of the dotted lines associated with the data class. Therefore with this intuition it is necessary to rewrite the constraint above to meet these new requisites:

$$y_i(< w, x_i > +b) \geq 1, \quad i = 1..N \tag{2.28}$$

Finally, all that is necessary is to maximize the distance between the dotted lines subject to the constraint set above.

Considering a hyperplane defined as $y(x_i) = 0$, where $y(x_i)$ takes the form of equation 2.23, the perpendicular distance of a point x_i to that hyperplane can be defined as $\frac{\|y(x)\|}{\|w\|}$ (see Figure 2.9) [8].

Furthermore, the required solutions are those in which all data points are correctly classified $y_i y(x_i) > 0$. Thus the distance of a point x_i to the decision surface is given by:

$$\frac{y_i y(x_i)}{\|w\|} = \frac{y_i(< w, x_i > +b)}{\|w\|} \tag{2.29}$$

Hence, the margin is given by the perpendicular distance to the closest point x_i from the data set, and the objective is to optimize the parameters w and b in order to maximize this

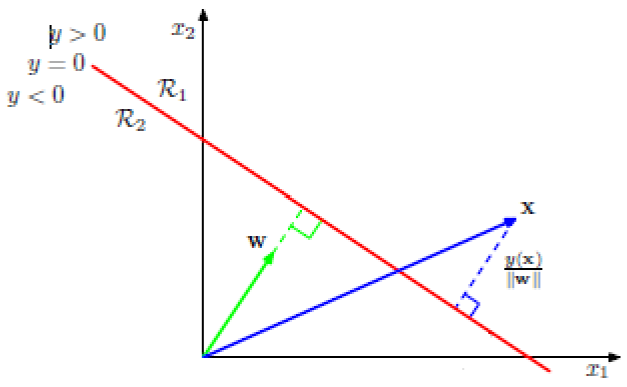


Figure 2.9: Distance of a point to the decision surface.

distance. Thus the maximum margin solution is defined as:

$$\arg\max_{w,b}\left\{\frac{1}{\|w\|}\underset{n}{\text{minimize}}\,[y_i(< w,x_i > +b)]\right\} \tag{2.30}$$

Combining equations 2.28 and 2.30, the optimization problem simple requires the maximization of $\|w\|^{-1}$, which is equivalent to minimize $\|w\|^2$. Therefore the optimization problem can be formulated as (the primal formulation):

$$\begin{aligned} &\arg\min_{w,b} \quad 1/2\,\|w\|^2 \\ &\text{subject to} \qquad\qquad y_i(< w,x_i > +b) \geq 1, \quad i = 1..N \end{aligned} \tag{2.31}$$

The factor 1/2 is included for computational convenience. This is an example of a quadratic programming problem in which the objective is to minimize a quadratic function subject to a set of linear inequalities constraints. As a first look, the parameter b (bias) has disappeared from the objective function. However, it is determined implicitly in the constraints, as changes in parameter w will be compensated with changes in parameter b.

The standard way to solve this constrained optimization problem is using Lagrange multipliers (a_i) and the Karush-Kuhn-Tucker (KKT) conditions, obtaining the dual formulation:

$$
\begin{aligned}
L(w,b,a) &= 1/2\,\|w\|^2 - \sum_i^N a_i \{y_i(<w,x_i> + b) - 1\} \\
\text{subject to} \qquad \partial L/\partial w &= 0 \Leftrightarrow w - \sum_i^N a_i y_i x_i = 0 \\
\partial L/\partial b &= 0 \Leftrightarrow \sum_i^N a_i y_i = 0 \\
a_i &\geq 0, \quad i = 1,..,N \\
a_i(y_i(<w,x_i> + b) - 1) &= 0, \quad i = 1,..,N \\
y_i(<w,x_i> + b) - 1 &\geq 0, \quad i = 1,..,N
\end{aligned}
\tag{2.32}
$$

Using these conditions the equation 2.31 be rewritten:

$$
\begin{aligned}
\arg\max_a \quad & \sum_{i=1}^N a_i - 1/2 \sum_{i,j}^N a_i a_j y_i y_j x_i^T x_j \\
\text{subject to} \quad & \sum_i^N a_i y_i = 0 \\
& a_i \geq 0, \quad i = 1..N
\end{aligned}
\tag{2.33}
$$

Going from the original primal formulation (equation 2.31), which has d variables (dimensionality of the input variables) to the dual formulation that has N variables (number of data points) seem disadvantageous, since in most of the cases $N >> d$. However, this formulation allows the model to be reformulated with kernels that will be explained later in this section.

After finding the optimal a (Lagrangian multipliers) the data points can be classified by evaluating:

$$
y(x) =< w,x > + b = \sum_i^N a_i y_i x^T x_i + b \tag{2.34}
$$

Note that, any training point with $a_i = 0$ plays no role in making predictions. The remaining data points are called the support Vectors that satisfy the condition $y_i y(x_i) = 1$ that result from the KKT conditions or from the very own formulation of SVM. For this reason, these points fall on the maximum margin hyperplane in feature space. Once the model is trained a significant amount of training data can be discarded, and only the support vectors are retained.

Having solved the quadratic problem and computed the values for the a variables (Lagrange Multipliers), now it is possible to compute the b (bias) variable.

Considering that as referred before for any support vector x_m the condition $y_m y(x_m) = 1 \Leftrightarrow y_m(< w, x_m > +b) = 1$ is respected, b can be computed by using equation 2.34:

$$y_m(\sum_i^N a_i y_i x_m^T x_i + b) = 1 \Leftrightarrow b = y_m - \sum_i^N a_i y_i x_m^T x_i \tag{2.35}$$

However, it is numerically more stable to average over ALL support vectors:

$$b = \frac{1}{N_s} \sum_{m \in S}^N \left\{ y_m - \sum_{i \in S}^N a_i y_i x_m^T x_i \right\} \tag{2.36}$$

With m and i running only over support vectors (S space).

At this moment it was only considered linear separable data. And what if the data is not linearly separable? If it is not possible to find a hyperplane that separates all of the input training data into the two corresponding classes correctly. In this case it will be impossible to find any combination of w and b that respect the constraints.

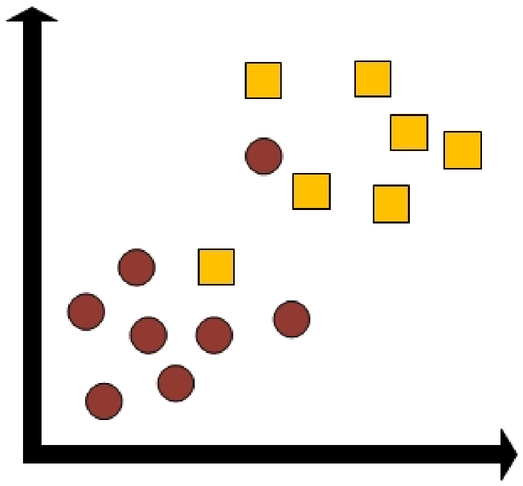


Figure 2.10: Non-linear classification problem.

For non linear separable data it is possible to find two situations:

- The non-linearly separable data is a result of noisy data. Therefore, keeping the linear separator and accept some error is more natural.
- The non-linearly separable data portraits some intrinsic property of the problem. In this case, a more complex classifier that allows more general boundaries between classes may be more appropriate.

Starting by considering the first case, the solution is to soft the constraints related with the data must fall in the correct side of the +1 and -1 hyperplanes. This will allow that some points to be misclassified.

Until now, it was presented and formulated the *hard margin SVM*, in reference to the fact that the margin constraints are hard, and are not allowed to be violated at all. Next it will be introduced the soft margin SVM, equations 2.37. This approach is probably the most used to balance the goal of maximum margin separation and correctness of the training classification.

$$\begin{aligned} &\underset{w,b,\xi}{\text{minimize}} && 1/2 < w,w > +C\sum_{i=1}^{N}\xi_i \\ &\text{such that} && y_i(w.x_i+b)+\xi_i \geq 1 \\ & \xi_i \geq 0 \text{for all} && i=1,2,\ldots,N \end{aligned} \tag{2.37}$$

First note that both the objective function and the constraint have a new variable (slack variable) ξ_i). The true meaning ξ_i is: instead of $y_i(< w,x_i > +b) \geq 1$, it is $y_i(< w,x_i > +b) \geq 1-\xi_i$, meaning that x_i is allowed to violate the margin by an amount of ξ_i.

Therefore, $\xi_i = 0$ for all data points that are on or inside the correct boundary margin and $\xi_i = |y_i - y(x_i)|$ for other points. Thus for data points that are on the decision boundary $(y(x_i) = 0)$ will have $\xi_i = 1$, and points with $\xi_i > 1$ will be misclassified (see Figure 2.11).

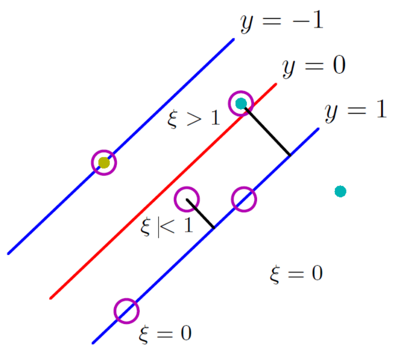


Figure 2.11: Slack variables values according to the positioning of the data in respect with the boundary.

Since, for any point misclassified $\xi_i > 1$, it makes it possible to consider $\sum_{i=1}^{N}\xi i$ as an upper bound of the number of misclassified points.

The parameter C controls the trade off between the dual objectives of maximizing the margin of separation and minimizing the misclassification error.

Therefore the objective is to minimize $1/2 < w,w > +C\sum_{i=1}^{N}\xi_i$ subject to the presented constraints plus the constraint $\xi_i \geq 0$. The corresponding Lagrangian function is given by:

$$L(w,b,\xi,\mu,a) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N} a_i\{y_i y(x_i) - 1 + \xi_i\} - \sum_{i=1}^{N}\mu_i\xi_i$$

subject to the KKT conditions

$$\begin{aligned} \mu_i &\geq 0 \\ y_i y(x_i) - 1 + \xi_i &\geq 0 \\ a(y_i y(x_i) - 1 + \xi_i) &= 0 \\ \mu_i &\geq 0 \\ \xi_i &\geq 0 \\ \mu_i\xi_i &= 0 \end{aligned} \tag{2.38}$$

Where a_i and μ_i are the Lagrange multipliers. Optimizing out w, b and ξ_i:

$$\begin{aligned} \frac{\partial L}{\partial w} &= 0 \Leftrightarrow w = \sum_{i=1}^{N} a_i y_i x_i \\ \frac{\partial L}{\partial b} &= 0 \Leftrightarrow \sum_{i=1} a_i y_i = 0 \\ \frac{\partial L}{\partial \xi_i} &= 0 \Leftrightarrow a_i = C - \mu_i \end{aligned} \tag{2.39}$$

Using these results the dual Lagrangian is obtained in the form:

$$\begin{aligned} \arg\max_a \quad L(a) &= \sum_{i=1} a_i - \frac{1}{2}\sum_{i,j=1}^{N} a_i a_j y_i y_j x_j^T x_i \\ \text{subject to} \quad & \sum_{i}^{N} a_i y_i = 0 \\ & 0 \leq a_i \leq C, \quad i = 1..N \end{aligned} \tag{2.40}$$

As presented before, the subset of data points with $a_i = 0$ do not contribute to the predictive model. Thus remaining data points constitute the support vectors, having $a_i > 0$ and satisfying $y_i y(x_i) = 1 - \xi_i$.

If $a_i < C$, than implies that $\mu_i > 0$, $\xi_i = 0$ and therefore such points lie on the margin. Points with $a_i = C$ can lie inside the margin and can either be correctly classified if $\xi_i \leq 1$ or misclassified if $\xi_i > 1$.

To determine the parameter b, note that support vectors for which $0 < a_i < C$ have $\xi = 0$. Therefore it is possible to write:

$$y_m(\sum_{i \in S}^{N} a_i y_i x_m^T x_i + b) = 1) \tag{2.41}$$

Again a more stable solution is obtained by averaging :

$$b = \frac{1}{N_M} \sum_{n \in M} (y_n - \sum_{i \in S}^{N} a_i y_i x_m^T x_i) \tag{2.42}$$

Where M is the set of data points having $0 < a_i < C$.

So far it was proved that the maximally separating hyperplane is a good starting point for linear classifiers. The formulation as an optimization problem for finding this hyperplane was then presented. Then, a way to deal with data that is not nearly separable was shown by allowing some training points to violate the margin.

In summary, until now it was only considered very simpler linear classifiers, that will perform well in "simpler" problems. Now, more complex classes of decision functions are to be introduced.

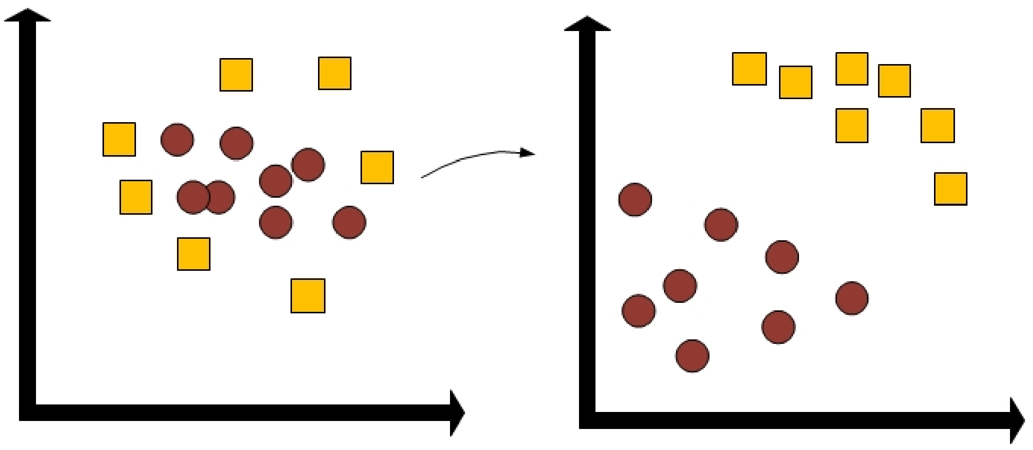


Figure 2.12: Change the feature space, by a kernel function ϕ in which the data becomes linearly separable.

In Figure 2.12, in the left side it is presented a case where the data is not close for linearly separable, and a solution based in a linear classifier will never perform well. Imagine, however, that exist a mapping ϕ which transforms the data space to a new one, possibly with a higher dimensional space, where it is possible to apply the linear SVM.

The changes in the problem formulation are minimum. Just replace $x_j^T x_i$ by $\phi(x_j)^T \phi(x_i) = k(x_i)k(x_j)$.

The non-linear SVM formulation now becomes:

$$\begin{aligned} \arg\max_a \quad & L(a) = \sum_{i=1} a_i - \frac{1}{2}\sum_{i,j=1}^{N} a_i a_j y_i y_j k(x_i)k(x_j) \\ \text{subject to} \quad & \sum_{i}^{N} a_i y_i = 0 \\ & 0 \le a_i \le C, \quad i = 1..N \end{aligned} \tag{2.43}$$

With,

$$b = \frac{1}{N_M}\sum_{n\in M}(y_n - \sum_{i\in S}^{N} a_i y_i k(x_i)k(x_j)) \tag{2.44}$$

And Prediction:

$$y(x) = \sum_{i=1}^{N} a_i y_i k(x, x_i) + b \tag{2.45}$$

The new feature space $\phi(x)$ (Kernel function) should capture the non-linearity intrinsic in the data set, not the noise. In its simple case $\phi(x) = x$ therefore $K(x_i, x_j) = x_i^T x_j$.

Some examples of kernel function are:

- Linear $k(x_1, x_2) = x_1^T x_2$
- Radial Basis Function or Gaussian $k(x_1, x_2) = exp(-\gamma \|x_1 - x_2\|^2)$. See Figure 2.13
- Polynomial $k(x1, x2) = (\gamma x_1^T x_2)^{degree}$. See Figure 2.13
- sigmoid: $tanh(\gamma x_1 x_2)$

Considering now multi-class classification problems, there are two main SVM-based approaches: *One against One* and *One against All*. The literature is inconclusive on the best approach to solve multi-class problems.

- One against all: This approach is used quite often for SVM training and prediction because it is only necessary to compute N models, where each model represents the classification for each class (belongs to a class or not). One of the biggest problems with this approach is the loss of significance in the dataset. A significant discrepancy between the number of observations may mislead or jeopardize the training process of SVM models.
- One against one: This method constructs one binary classifier for every pair of distinct classes, meaning that all together are $N(N+1)/2$ binary classifiers. After

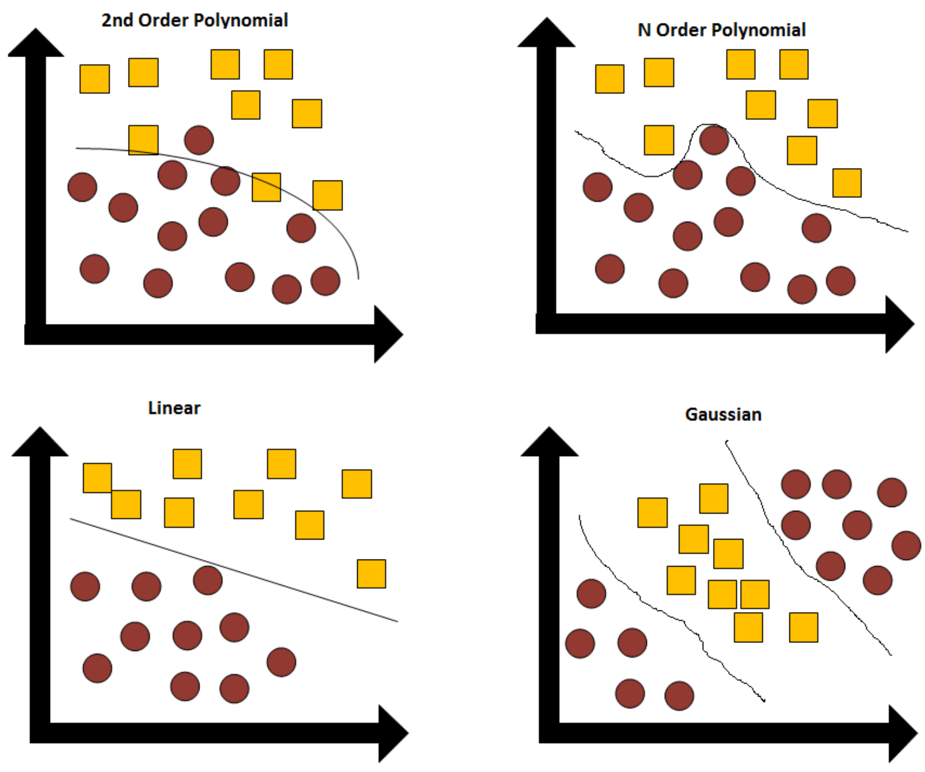


Figure 2.13: Example of Linear Guassian and Polynomial Kernel functions applied to some data.

the vote of each of the $N(N+1) = 2$ binary classifiers, the final result can be the class with the largest number of votes. Other approach is the iterative exclusion of the class that loses, i.e., all SVM models belonging to this class versus those that can be excluded from the classification process, where only the models of the victorious class should be considered. Finally, there is a tree type-like approach where the classification result is the strongest class.

In conclusion, the disadvantage of using SVM is the lack of transparency in the results, as it cannot represent the score of all the classes as a simple function of the input features, since its dimension can be very high [4]. Another disadvantage is the SVM parameter tuning that in general is not trivial to perform.

2.5 Classifier Model Selection and Assessment

As it is possible to conclude from the last section, machine learning models tuning must be performed. Moreover, to overcome the problem of over-fitting the data, model selection and assessment strategies are usually applied.

- Model Selection: estimating the performance of different models (different parameter set-up) in order to choose the best one.
- Model Assessment: having chosen a final model, estimating its generalization error on new independent data.

Therefore, to accomplish the model selection and assessment, the dataset is split into two parts. Training set: used to fit the models and to determine prediction error for model selection, and the test set: used to assess the generalization error for the tuned model. In classification problems, the main methods for model selection and assessment are Leave-one-out, Cross-validation with random sub-sampling and K-fold cross-validation. In K-fold cross-validation, the training data, initially obtained by diving the data recorded into train and test data, is divided into k subsets, Figure 2.14.

At each time, k-1 subsets are put together to form the training set, and the remaining one is used as the validation set in order to compute the misclassification error in the training phase. Usually, this misclassification error will be less than the generalization error, and to estimate the last one the model needs to applied to independent test samples.

The big advantage of this method is that every data point gets to be in the validation set exactly once and gets to be in a training set k-1 times. This procedure makes it possible to try to avoid the over-fitting of the models. And at the same time achieve a more reliable models performance evaluation.

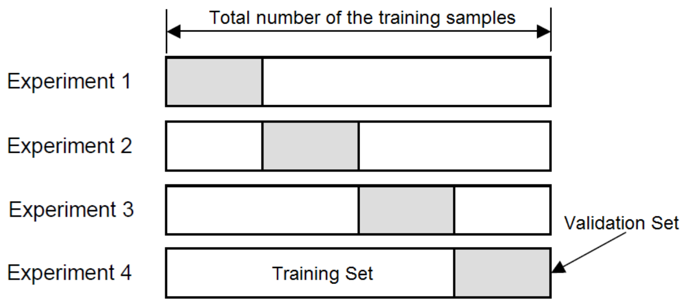


Figure 2.14: K-fold Cross-validation scheme.

However, the disadvantages of this method are that the training algorithm has to be rerun from scratch k times, which means that it takes k times and as much computational cost to make a single evaluation.

For the problem here presented, the K-fold cross-validation algorithm was the selected one, since, with this technique, all the training data is considered for estimating the accuracy value during this phase. This will result, in a better estimation of the performance of the model in new data.

2.6 Pattern Recognition based on Machine Learning Models - Final Architecture

The proposed classification architecture can be divided into two phases: the training phase and the production phase.

2.6.1 Training Phase

During the training phase, all the procedures presented in this Chapter are used, see the flow chart presented in Figure 2.15.

In the beginning K models of each of type of parts are considered. These are the entrance of the flow chart. From these K models features are extracted, being organized in two subsets $F_{train}(n \times m)$ and $F_{test}(i \times m)$, where n and i denotes the number of models in each subset and m is the number of features extracted.

Beyond that, there exists a vector; $s_{1 \times k}$ that indicates which features are active or not (k represent the total number of active features). In this way, the simulated annealing in each iteration will vary the active features, training the classifier model (k-Nearest Neighbour, Neural Network or Support Vector Machine), tune their parameters using K-fold cross-validation (in k-Nearest Neighbour the k value, in Neural Networks, for example, the number of hidden layers considered, etc) and test the classification models. With this procedure, the author intend to eliminate the features that do not contribute or even jeopardize the classifier model, tune the parameters of each classification model and avoid models data over-fitting.

In the end a subset of features and the best model classification parameters that minimize the classification error (percentage of incorrect classification parts), will be driven. A model of the classification algorithm with these parameters will be trained and tested in an independent test samples, obtaining in this way the expected generalization error for the object recognition model.

The classifier model is stored, as well as the vector of active features, $s_{1 \times k}$, since they will be used in the production phase for feature selection and classification purposes.

2.6.2 Production Phase

For the production phase, and having extracted the classifier model in the training phase, it stands simply to:

- Extract the 3D model of the unknown part.

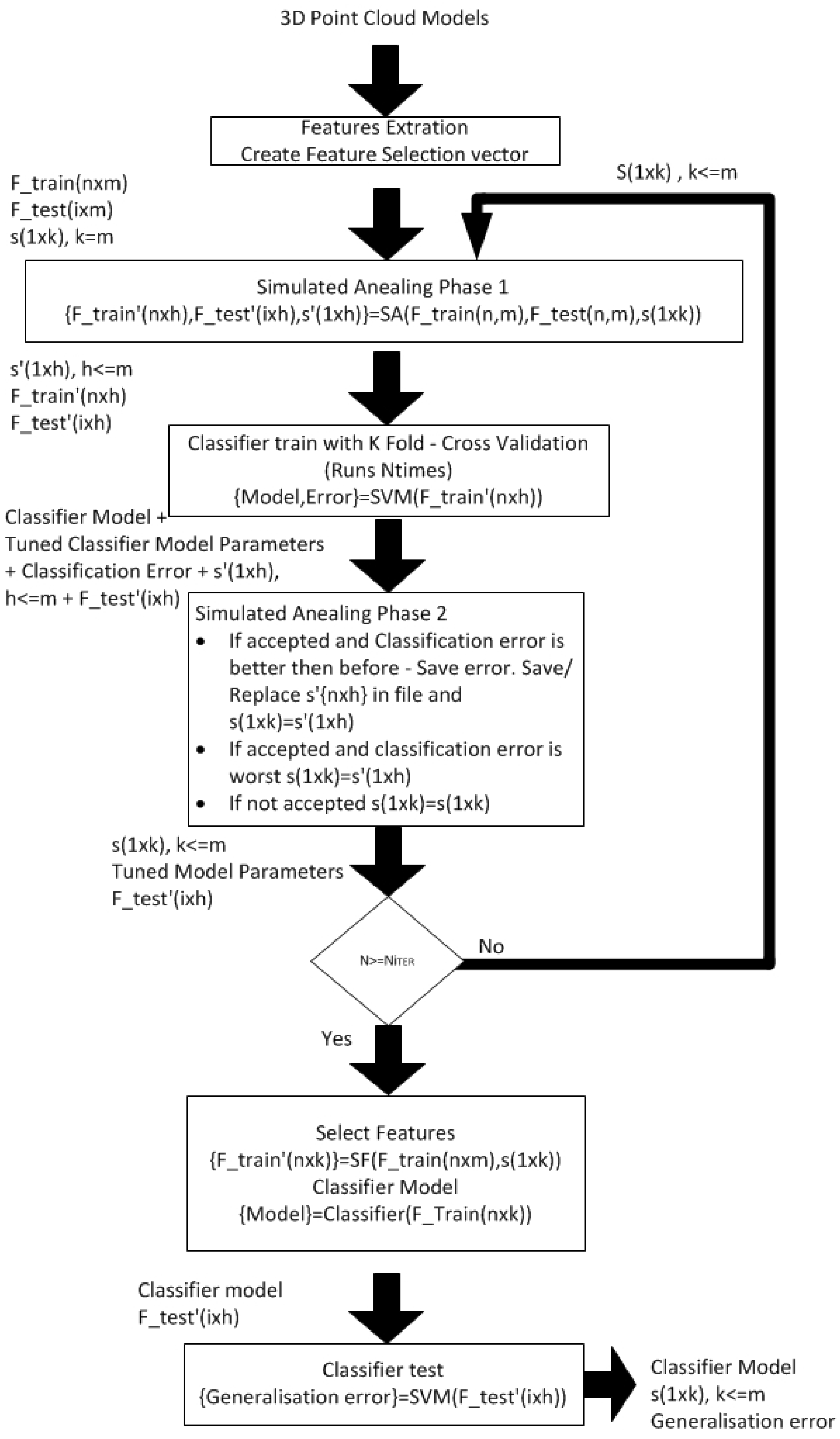

3D Point Cloud Models
Features Extration
Create Feature Selection vector
F_train(nxm)
F_test(ixm)
s(1xk), k=m
S(1xk) , k<=m
Simulated Anealing Phase 1
{F_train'(nxh),F_test'(ixh),s'(1xh)}=SA(F_train(n,m),F_test(n,m),s(1xk))
s'(1xh), h<=m
F_train'(nxh)
F_test'(ixh)
Classifier train with K Fold - Cross Validation
(Runs Ntimes)
{Model,Error}=SVM(F_train'(nxh))
Classifier Model +
Tuned Classifier Model Parameters
+ Classification Error + s'(1xh),
h<=m + F_test'(ixh)
Simulated Anealing Phase 2
If accepted and Classification error is better then before - Save error. Save/ Replace s'{nxh} in file and s(1xk)=s'(1xh)
If accepted and classification error is worst s(1xk)=s'(1xh)
If not accepted s(1xk)=s(1xk)
s(1xk), k<=m
Tuned Model Parameters
F_test'(ixh)
N>=NITER
No
Yes
Select Features
{F_train'(nxk)}=SF(F_train(nxm),s(1xk))
Classifier Model
{Model}=Classifier(F_Train(nxk))
Classifier model
F_test'(ixh)
Classifier test
{Generalisation error}=SVM(F_test'(ixh))
Classifier Model
s(1xk), k<=m
Generalisation error

Figure 2.15: Support Vector Machine - Train flow chart.

- Feature Extraction - considering the features only selected by the simulated annealing phase ($s_{1\times k}$).
- Run the classifier model.
- The result of the classifier in this case is a probabilistic vector that indicates the percentage of the unknown model to fit in the classes previously trained.

2.7 The Perfect Match (PM) - Point Cloud Matching and Pose Estimation

As refereed in the begin of this Chapter, the use of PM algorithm was also considered for object recognition and pose estimation. The idea is to compare the 3D model of the unknown part with previously recorded ones. The matching with the least error value will be the class of the unknown part.

The presented approach do not rely in the extraction of features from the objects, but in the direct comparison of the object model. With this, a different approach, when comparing with SVM, is to be explored hoping to introduce some intrinsic model information that it is not possible to be capture by the feature extraction step. Note that the PM algorithm was not applied to all the classes of objects recorded in database, since it would become computationally heavy.

To perform this matching the algorithm presented in [34] and more recently adapted for 3D Matching by Miguel Pinto [56] was used.

2.7.1 Perfect Match using 2D data - Robot Soccer Field Pose Estimation

The algorithm presented in [34] is nowadays used or adapted by some of the worlds soccer robotic teams (middle size and NAO's) in the robots localization on the soccer field, by using an estimation of their distance to the closest lines, through the use of CCD/CMOS cameras in the robot structure [59, 79, 64]. High precision, robustness and computational efficiency are some of the motives which make this algorithm so used in the robotic soccer field.

In their application scenario, the objective is to estimate the position and heading of the robot with respect to the information retrieved by the processed image (field markings) by using an error function that describes the fitness of a certain estimate to a soccer field map previously recorded.

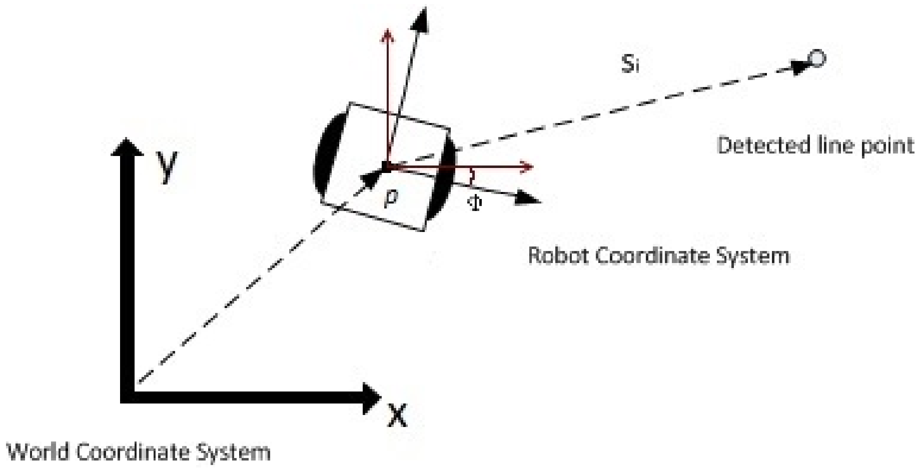


Figure 2.16: Sketch of the world coordinate system, the robot relative coordinate system and a vector s_i pointing to a detected line point.

Therefore, letting (p,ϕ) be a pair of possible robot positions $p=(px,py)$ and heading ϕ in a global coordinated system and s_i be the vectors list of detected line points, the line position in the world coordinates is given by $p+\begin{pmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{pmatrix} * s_i$, see Figure 2.16.

Minimizing the error between detected line points and true field marking means to solve:

$$\underset{p,\phi}{\text{minimize}} \quad E=\sum_{i=1} err(d(p+\begin{pmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{pmatrix}) * s_i)) \tag{2.46}$$

where E represents the matching error of the soccer field lines estimated pose (estimated by the robots view) and the matching with the true position of the robotic map saved - sum of the distance of the computed line point positions (using the robot camera) with the soccer field map saved in a database.

$d(.)$ gives the distance from a certain point on the field to the closest field marking.

err is the error function that punishes deviations between detected lines and the field model. The most used error function is $1/2*d(.)^2$ that is standard in many applications although is not appropriate for the task since is not robust to outliers. The selected error function used is $1-c^2/(c^2+d(.)^2)$ (see Figure 2.17 and reference [34] for more detailed information).

Due to the non-linearity of the minimization problem it is not possible to analytically calculate its solution so a numerical minimizer is needed. Since $d(.)$ is almost everywhere differentiable it is possible to build its gradient almost everywhere and interpolate the gradient at the non-differentiable places. Hence, gradient descent algorithm is used to solve the minimization problem. Due to RPROP quick convergence and high robustness this algorithm was the selected one to solve the minimization task.

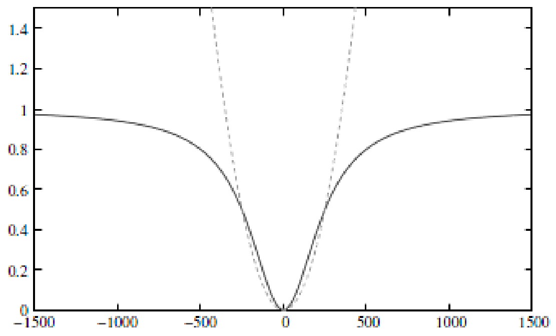


Figure 2.17: Comparison between the squared error function (dashed line) and the used in this algorithm (solid Line).

For computational performance purposes, the distance map and respective gradient along the x axis and y axis are computed off-line. This feature is one of the most important for computational efficiency of the algorithm.

2.7.2 Perfect Match Using 3D Data

Miguel Pinto et al., in [56] updates the PM algorithm to 3D matching, although the final result estimation robot pose persists. The main difference is that their approach is based on a point cloud retrieved by an LRF and its match in previously computed scene map. In other words, in the referred article, the authors started by acquiring a 3D Map of the environment with a laser range finder coupled to the mobile robot. After the creation of this 3D Map and considering off-line mode, a distance map and a gradient map are created and stored. The stored distance and gradient matrices are then used as look-up tables for the 3D matching procedure, in the on-line robot motion. For the creation of the distance matrix, the distance transform is applied in the 3D occupancy grid of the world map. Furthermore, the Sobel Filter, again in 3D space, is applied to obtain the gradient matrices, in both x and y direction.

2.7.3 Matching Algorithm Applied to 3D Object Models - Classification Purpose

Making now the parallel to the application presented in this thesis, the 3D Map is the 3D model of each of the parts to be recognized. Therefore, for each of these models it is necessary to store equivalent 3D distance according to the world grid map, and the gradient matrices along x (width) and y (height) axis of the part. Note that, only the gradient along x and y axis will be computed since the PM is only using 3 degree of freedom (3 Dof - x, y and ϕ).

2.7.3.1 Creating the 3D Model, Distance and Gradient Matrices

For the computation of the 3D world occupancy grid map, where the 3D part model is recorded, a limit of y pixels along the height of the part and a maximum height of *maxheight* was considered, achieving a resolution of *res* mm. Considering a maximum width *maxwidth* and a maximum depth of *maxdepth* of the part the world grid map is a 3D matrix of $\frac{maxwidth}{res} \times \frac{maxheight}{res} \times \frac{maxdepth}{res} cells$. See Figure 2.18.

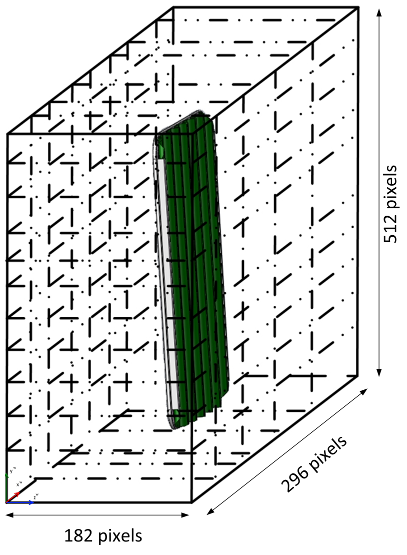


Figure 2.18: 3D World occupancy Grid Map.

After the construction of the 3D part model, in this world grid map, the distance and the gradient matrices need to be computed. These matrices have equal dimension and resolution of the world occupancy grid map.

The distance matrix, represented by *DistMap*, has at each coordinate position in the world reference frame the distance to the nearest 'occupied cell', in the occupancy grid. For the computation of this matrix, it is necessary to initialize it with zeros in the position where the cells in the 3D world occupancy grid map are occupied and in the others with a very large number value. Then a standard procedure is applied. For more information refer to the work presented by Stefano Carpin et. al [9] or Miguel Pinto [56]. Figure 2.19 illustrates the final result. For more information refer to the work [9].

Having now defined the distance matrix, now it is possible to compute the gradient matrix along x (∇x_{3D}) and y (∇y_{3D}). This corresponds to the variation of the *DistMap* in each world position with the variation of x and y correspondingly.

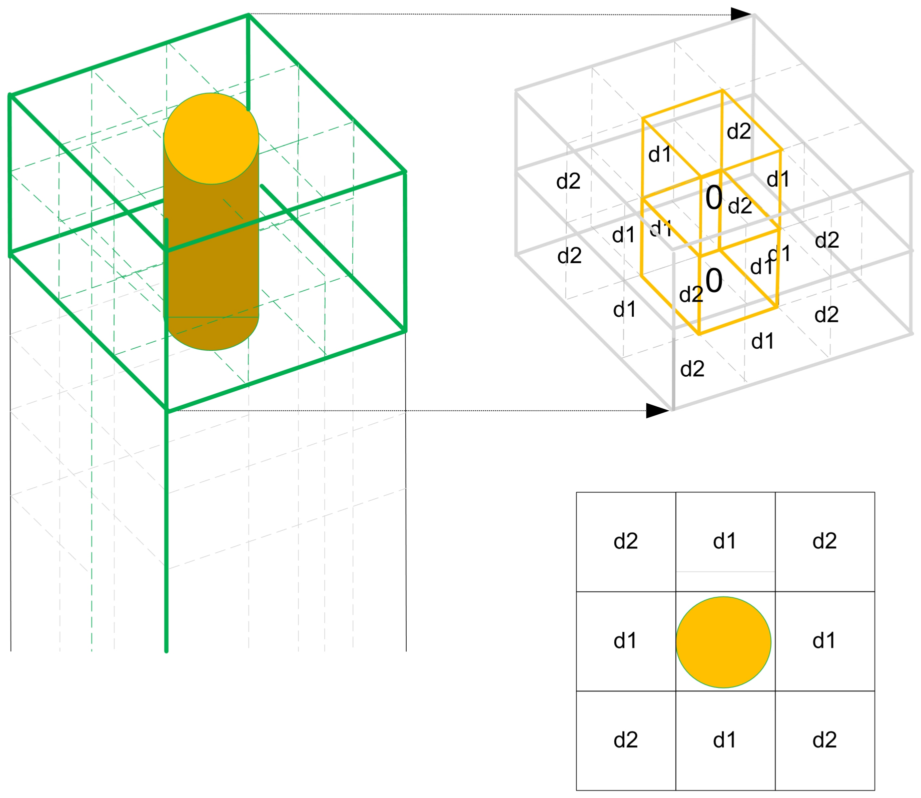


Figure 2.19: Final result of the distance matrix computation procedure.

The computation of ∇x_{2D} is equal to the weighted average of $DistMap$ using a vertical Sobel filter (equation 2.47). In the same way, ∇y_{2D} is equal to the weighted average of $DistMap$ but in this time using a horizontal Sobel filter (equation 2.47).

$$H = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{2.47}$$

$$V = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{2.48}$$

In this way and considering that for each position p with coordinates in the world reference frame $[x^W, y^W, z^W]$ and with distance value equal to $DistMap(p) = dist$, the neighbours are defined as:

$$N(x^W, y^W, z^W) = \begin{bmatrix} lo & mo & ro \\ lm & dist & rm \\ lu & mu & ru \end{bmatrix} \tag{2.49}$$

Therefore the ∇x_{2D} and ∇y_{2D} are computed from:

$$\nabla x_{2D}[x^W, y^W, z^W] = \frac{H.N(x^W, y^W, z^W)}{8} \tag{2.50}$$

$$\nabla y_{2D}[x^W, y^W, z^W] = \frac{V.N(x^W, y^W, z^W)}{8} \tag{2.51}$$

Remember that the problem presented is in 3D space therefore:

$$\nabla x_{3D}[x^W, y^W, z^W] = \frac{\nabla x_{2D}[x^W, y^W, z^W - 1] + \nabla x_{2D}[x^W, y^W, z^W] + \nabla x_{2D}[x^W, y^W, z^W + 1]}{3} \tag{2.52}$$

$$\nabla y_{3D}[x^W, y^W, z^W] = \frac{\nabla y_{2D}[x^W, y^W, z^W - 1] + \nabla y_{2D}[x^W, y^W, z^W] + \nabla y_{2D}[x^W, y^W, z^W + 1]}{3} \tag{2.53}$$

2.7.3.2 State Variables

Another important parallels that it necessary to be made is the variables present in the localization and the matching problems. Therefore, for mobile robotic purpose the objective is to estimate the pose (x, y position and orientation) of the robot in the 2D Map. For the object recognition cases beyond the problem of identifying the model with the smallest matching error, with this approach it will be possible to estimate the displacement x and y and orientation (along z axis - depth) of the new model when compared with the stored one (state X_{Match}).

Note that it is assumed that all the 3D model points are in the world referential.

$$X_{Match} = [x_{Match} \;\; y_{Match} \;\; \theta_{Match}] \tag{2.54}$$

In this sense, for the stored models and since they are considered as the reference ones X_{Match} will be equal to $[0,0,0]$. Now, consider a list of 3D model points (unknown Model) with points $[x_i^L, y_i^L, z_i^L]$. This unknown model will be compared with the reference ones and its displacement (X_{Match}) will be computed.

Therefore, it is possible to write:

$$\begin{bmatrix} x_i^{Lnew} \\ y_i^{Lnew} \\ z_i^{Lnew} \end{bmatrix} = \begin{bmatrix} x_{Match} \\ y_{Match} \\ 0 \end{bmatrix} + \begin{bmatrix} cos\theta_{Match} & -sin\theta_{Match} & 0 \\ sin\theta_{Match} & cos\theta_{Match} & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_i^L \\ y_i^L \\ z_i^L \end{bmatrix} \tag{2.55}$$

The 3D PM runs in the steps presented in algorithm 1, and it can be divided into two phases.

```
Data: $X_{Match} \leftarrow X_{init}$
      $\nabla E(0) \leftarrow 0$
for $n \leftarrow 1$ $to$ $iterMax$ do
    $X^{Lnew} \leftarrow X_{Match} + R \cdot X^L$
    $\nabla E(n) \leftarrow GradientMatrix(X^L new, X_{Match})$
    $X_M atch \leftarrow RPROP(\nabla E(n), \nabla E(n-1), X_{Match}$
end
Result: $X_{Match}$ $and$ $E(Matching$ $Error)$
```

Algorithm 1: Pseudo-Code for the Perfect Match algorithm.

2.7.3.3 Phase 1 - Computation of the matching error

The distance matrix, stored in memory, is used to compute the matching error (E). The matching error is computed through the cost value of the list of 3D model points changed $[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}]$:

$$E = \sum_{i=1}^{N} E_i, \qquad E_i = 1 - \frac{L_c^2}{L_c^2 + d_i^2} \tag{2.56}$$

Where d_i and E_i are representative of the distance matrix ($DistMap$) and the cost function E for the 3D Model Points $[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}]$ respectively. N is the number of points in the 3D Model. L_c is an adjustable parameter that controls the contribution of the outliers for error estimation.

2.7.3.4 Phase 2 - Optimization routine Resilient Back-Propagation (RPROP)

For the computation of the RPROP the gradient of the cost function in order to the X_{Match} needs to be computed, and it is given by:

$$\begin{aligned}
\nabla E(k) = [\nabla E_x \;\; \nabla E_y \;\; \nabla E_\theta] &\iff \\
\nabla E(k) = \left[\frac{\partial E}{\partial x_{Match}} \;\; \frac{\partial E}{\partial y_{Match}} \;\; \frac{\partial E}{\partial \theta_{Match}}\right] &\iff \\
\nabla E(k) = \left[\sum_{i=1}^{N}\frac{\partial E_i}{\partial x_{Match}} \;\; \sum_{i=1}^{N}\frac{\partial E_i}{\partial y_{Match}} \;\; \sum_{i=1}^{N}\frac{\partial E_i}{\partial \theta_{Match}}\right] &\iff \\
\nabla E(k) = \sum_{i=1}^{N}\frac{2\cdot L_c^2\cdot d_i}{(L_c^2+d_i^2)^2}\left[\frac{\partial d_i}{\partial x_{Match}} \;\; \frac{\partial d_i}{\partial y_{Match}} \;\; \frac{\partial d_i}{\partial \theta_{Match}}\right] &\iff \\
\nabla E(k) = \sum_{i=1}^{N}\frac{2\cdot L_c^2\cdot d_i}{(L_c^2+d_i^2)^2}\frac{\partial d_i}{\partial X_{Match}} &
\end{aligned} \tag{2.57}$$

Where $\frac{\partial d_i}{\partial X_{Match}}$ can be re-writen:

$$\frac{\partial d_i}{\partial X_{Match}} = \left[\frac{\partial d_i}{\partial x_i^{Lnew}}\cdot\frac{\partial x_i^{Lnew}}{\partial x_{Match}}, \frac{\partial d_i}{\partial y_i^{Lnew}}\cdot\frac{\partial y_i^{Lnew}}{\partial y_{Match}}, \frac{\partial d_i}{\partial x_i^{Lnew}}\cdot\frac{\partial x_i^{Lnew}}{\partial \theta_{Match}} + \frac{\partial d_i}{\partial y_i^{Lnew}}\cdot\frac{\partial y_i^{Lnew}}{\partial \theta_{Match}}\right]^T \tag{2.58}$$

Analysing now each partial derivative, and by observing equation 2.54, it is possible to write:

$$\frac{\partial x_i^{Lnew}}{\partial x_{Match}} = 1, \qquad \frac{\partial y_i^{Lnew}}{\partial y_{Match}} = 1 \tag{2.59}$$

$$\frac{\partial d_i}{\partial x_i^{Lnew}} = \nabla x_{3D}[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}], \qquad \frac{\partial d_i}{\partial y_i^{Lnew}} = \nabla y_{3D}[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}] \tag{2.60}$$

$$\frac{\partial x_i^{Lnew}}{\partial \theta_{Match}} = -sin\theta_{Match}\cdot x_i^{Lnew} - cos\theta_{Match}\cdot y_i^{Lnew} \tag{2.61}$$

$$\frac{\partial y_i^{Lnew}}{\partial \theta_{Match}} = cos\theta_{Match}\cdot x_i^{Lnew} - sin\theta_m\cdot y_i^{Lnew} \tag{2.62}$$

Note that the ∇x_{3D} and ∇y_{3D} are the gradient values of the computed and stored 3D matrices, at the position $[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}]$. In this sense the partial derivative vector $\frac{\partial d_i}{\partial X_{Match}}$ in equation 2.57 can be re-written as:

$$\frac{\partial d_i}{\partial x_{Match}} = \nabla x_{3D}[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}] \tag{2.63}$$

$$\frac{\partial d_i}{\partial y_{Match}} = \nabla y_{3D}[x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}] \tag{2.64}$$

$$\frac{\partial d_i}{\partial \theta_{Match}} = \begin{bmatrix} \nabla x_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \\ \nabla y_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \end{bmatrix}^T \begin{bmatrix} -sin\theta_{Match} & -cos\theta_{Match} \\ cos\theta_{Match} & -sin\theta_{Match} \end{bmatrix} \begin{bmatrix} x_i^{Lnew} \\ y_i^{Lnew} \end{bmatrix} \tag{2.65}$$

In summary, the matching error between the two models of the part can be computed by the equation presented in 2.56. The gradient matrix about the cost function in order to the state of the model (X_{Match}) is given by the vector:

$$\nabla E(k) = \sum_{i=1}^{N} \frac{2 \cdot L_c^2 \cdot d_i}{(L_c^2 + d_i^2)^2} \begin{bmatrix} \nabla x_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \\ \nabla y_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \\ \begin{bmatrix} \nabla x_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \\ \nabla y_{3D}(x_i^{Lnew}, y_i^{Lnew}, z_i^{Lnew}) \end{bmatrix}^T \begin{bmatrix} -sin\theta_{Match} & -cos\theta_{Match} \\ cos\theta_{Match} & -sin\theta_{Match} \end{bmatrix} \begin{bmatrix} x_i^{Lnew} \\ y_i^{Lnew} \end{bmatrix} \end{bmatrix} \tag{2.66}$$

Reminding that ∇x_{3D} and ∇y_{3D} (gradient matrices) were computed and stored in the beginning of the algorithm, they are then used as look-up table; therefore, the computational efficiency of this procedure is very high.

After computing the matching error, RPROP is applied to each model variable. The RPROP routine can be described as the following: the next steps presented are performed to each variable that is intended to estimate, x_{Match} , y_{Match} and θ_{Match} (The steps are presented for the x reference frame, but it is identical for the other dimensions).

For the actual partial derivative $\frac{\partial E(k)}{\partial x_{Match}}$:

1. if $\frac{\partial E(k)}{\partial x_{Match}} - \frac{\partial E(k-1)}{\partial x_{Match}}$ is greater than zero, it means that the algorithm is converging in the right direction, matching error E minimization. In this case, the RPROP step $\lambda_{x_{Match}}$ will be increased accelerating in this way the convergence:

$$\lambda_{x_{Match}} = \lambda_{x_{Match}} \times \beta^+_{x_{Match}}, \quad \beta^+_{x_{Match}} > 1 \tag{2.67}$$

2. if $\frac{\partial E(k)}{\partial x_{Match}} - \frac{\partial E(k-1)}{\partial x_{Match}}$ is lower than zero, it means that the algorithm as passed a local minimum and the direction of convergence needs to be inverted. In this case, the RPROP step $\lambda_{x_{Match}}$ will be decreased to decelerate the convergence:

$$\lambda_{x_{Match}} = \lambda_{x_{Match}} \times \beta^-_{x_{Match}}, \quad \beta^-_{x_{Match}} \in]0,1[\tag{2.68}$$

3. If $\frac{\partial E(k)}{\partial x_{Match}}$ is higher than zero it means that the cost function E is increasing in the positive direction of x_{Match}. In this sense ,x_{Match} should be decreased in the amount of the step $\lambda_{x_{Match}}$.

$$x_{Match} = x_{Match} - \lambda_{x_{Match}} \tag{2.69}$$

4. If $\frac{\partial E(k)}{\partial x_{Match}}$ is lower than zero it means that the cost function E is decreasing in the positive direction of x_{Match}. In this sense x_{Match} should be increased in the amount of the step $\lambda_{x_{Match}}$.

$$x_{Match} = x_{Match} + \lambda_{x_{Match}} \tag{2.70}$$

In the end, a new estimation for X_{Match} (displacement of the actual 3D Model with the one recorded in the database) is computed, and the algorithm runs over again from Phase1 until the number of iteration iterMax is achieved.

The limitation of the number of iterations makes it possible to guarantee a maximum time of execution for this algorithm.

2.7.4 Perfect Match Training and Production

The PM algorithm can be divided into two steps: Teaching and Production Phase, Figure 2.20

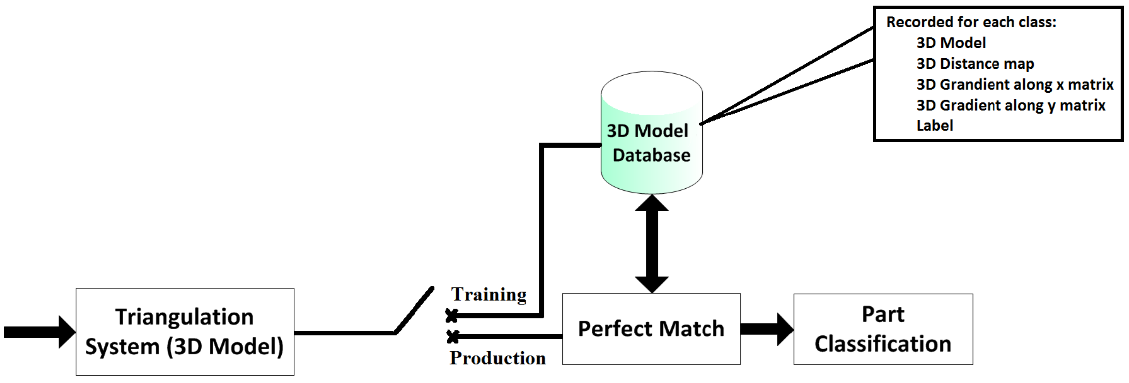


Figure 2.20: Perfect Match architecture.

The teaching phase consists of acquiring a 3D model of the new object that is to be introduced in the production chain. Therefore, the operator will insert this part type in the production line and the developed system will capture/store is 3D model and compute the distance map and gradient maps necessary for the matching algorithm. This procedure only needs to be performed once for each type of part. In the end, a database with the trained parts and respective names is dynamically created.

For the production phase, the operator only needs to insert the already trained part type in the production line. Then and by using the PM algorithm presented before the model of the unknown part will be compared with the one in the database, retrieving its classification. Then, this classification information is transferred to the industrial robot and the correct program is uploaded. If the operator insert a part that it was not yet trained two situations may occur: by evaluating the magnitude of the error it is considered that the part is not recognizable, or it is misclassified.

Beyond parts identification, its pose displacement in comparison with the one in the database will also be computed. Therefore, and assuming that the industrial robot was taught to manipulate perfectly the models saved in the database, it will be possible to send to the robot the trajectory adjustments along x (width), y (height) and rotation along z (depth) axis.

Chapter 3

Technologies for Feature and Data Acquisition

To apply the object recognition and pose estimation algorithms presented in the last Chapter, it is necessary a previously recorded 2D/3D model of the object to be recognized. Therefore, a state-of-art revision, a comparison (analysing the advantages and disadvantages) and a selection of the modelling sensors will be performed in this Chapter.

3.1 Related Work on 3D Model Reconstruction Sensors

In the last decade, the demand for 3D modelling sensors has increased. In part, as a result of the software development verified on 3D data processing but also in terms of hardware, where the development verified at electronic equipments, mechanical sensors structure and computational power contributed for the 3D modelling solutions widespread in different fields.

The robotics area is not insensible to these developments, where more and more solutions based on 3D data for different purposes (image registration, mobile robot localization, obstacle avoidance, and others) are becoming available.

In [68] it is presented a reasonable set of three dimensional image reconstruction techniques. Among them, it is possible to highlight: Laser Triangulators, Structured light, Stereo Vision, Photogrammetry and Time-of-Flight.

3.1.1 Laser Triangulators

Laser Triangulation systems are nowadays the most common non-contact method in industrial equipment like coordinate measure machines [69](mainly due to the system precision values) being also considered a fast computational technique with a large range of applications [1].

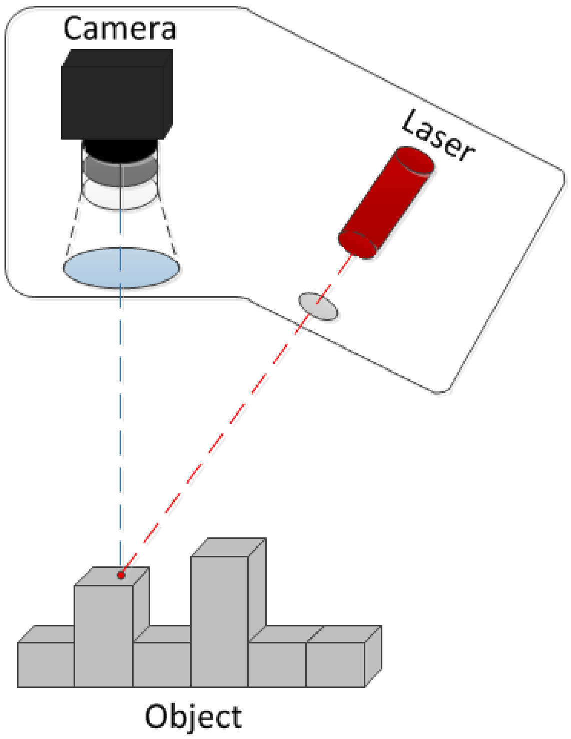


Figure 3.1: Laser Triangulation concept.

In Laser Triangulators category it is possible to find single point (see Figure 3.1) and laser stripes triangulators (additional lens that expands the light beam along one direction creating a plane of light). In their initial set-up, a compromise is necessary between the field of view, the measurement resolution and the shadow effect imposed by the object shapes that are directly related with the angle between the laser and camera. Furthermore, the system measurement precision value is directly related to the width of the laser line, the camera resolution and the laser pose (translation and orientation) relatively with the camera.

Considering this 3D modelling system, Miguel Pinto et. al in [55] sense and measure the position, rotation and dimensions of a cork piece in a conveyor belt in a simplistic scenario. The good performance of the modelling system allowed the computation of the object rotation angle in the axis normal to the conveyor plane. This procedure allowed the performance of object grasping by an industrial robot.

Using the same modelling system, Marcos Ferreira et al. in [15] present an object recognition work, this time in a painting application. One more time good 3D object models with low level of noise were obtained.

For its turn, Dejan Seatovic in [72] developed an automatic recognition and plant-treatment system based in an infra-red laser triangulation sensor and in a high-resolution smart camera.

Just to conclude, the main advantages of laser triangulators are their accuracy and their relative insensitivity to light conditions and surface texture effects. The major disadvantage are: the need to create object linear movement so the 3D model can be captured; the need of structured environment to limit environments light changes; and also object occlusions due the angle between the camera and the laser.

3.1.2 Structured Light

Structured Light based sensors share the same principle as the laser triangulators. The main difference is, instead of scanning the surface, a 2D pattern of non-coherent light is projected into the object, see Figure 3.2 [1]. Then the range information is obtained simultaneously by analysing the pattern deformation [68].

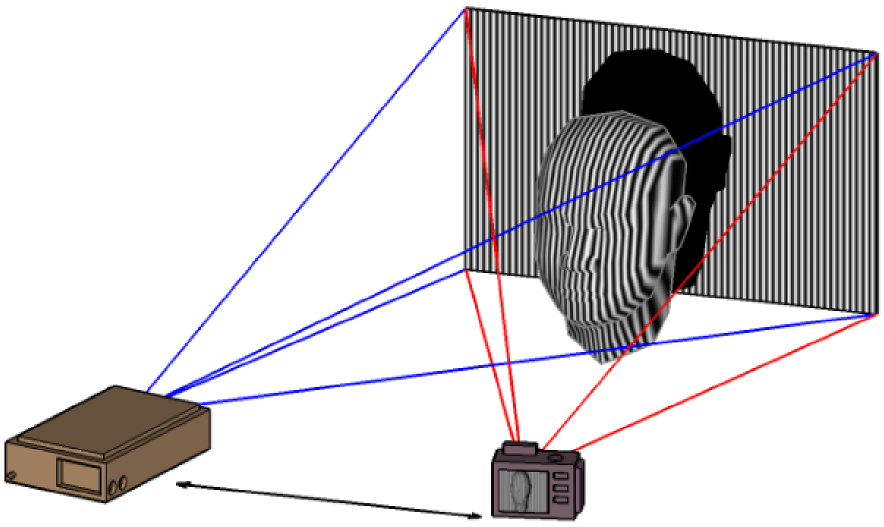


Figure 3.2: Structured Light concept.

The major problem related with this approach is to guarantee that different object points are assigned to the correct projected pattern plane [68]. To overcome this problem, many projection strategies have been developed: grid patterns [35], dot patterns [45],

[1]Image from Alce Technology - 3D depth capture and Structured Light

multi vertical slits [42] and fringe patterns [67]. Beyond that, the general performance of the method depends on the environment light. Note also that, structured light normally represent a higher investment when compared with the laser triangulation approach.

Using a very similar concept, nowadays Kinect [2] receives most of the attentions for 3D modelling, due to its low cost value. Although it has high potential (due to the capacity of extracting 3D Points clouds adding the colour feature) its resolution falls short when comparing with other low cost solutions.

Following this approach, the authors in [23] used Kinect to perform the segmentation of objects present in the scene. They utilize the Kinect RGB camera to perform object colour segmentation, and the depth information to discriminate objects that are not in the same plane of interest.

3.1.3 Stereo Vision

In the stereo vision approach, two or more cameras are used to concurrently capture the same scene. To perform the 3D reconstruction the identification of common points in each pair of images is necessary (triangulation - see Figure 3.3), which can become a difficult task for complex scenarios.

As an alternative for this approach, it is possible to use one camera and capture image scenes from different viewpoints. However, the object of interest must not have "unknown" movement [68].

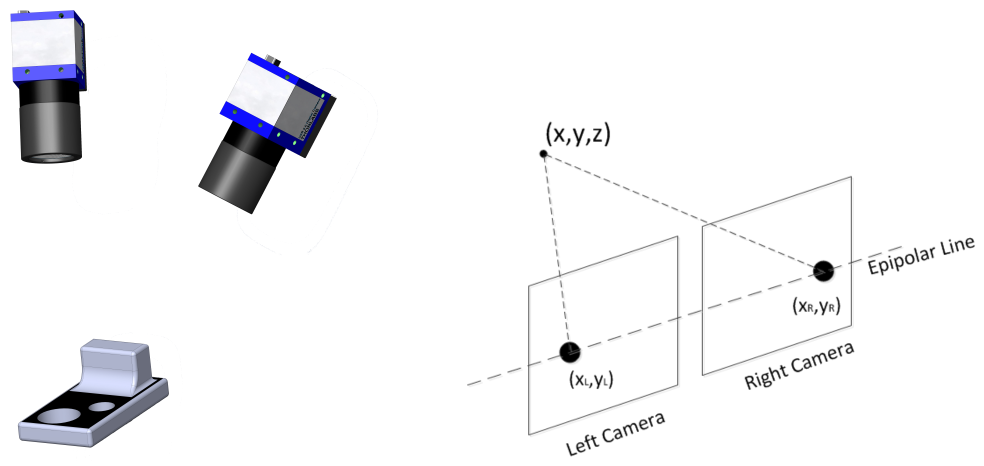


Figure 3.3: Stereo Vision concept.

Using stereo vision Sergiu Nedevschi et al. [46] presents an object detection system that uses 3D information provided by stereo reconstruction. According to the authors the

[2]http://www.microsoft.com/en-us/kinectforwindows/

resulting system is a high-accuracy, far distance obstacle detector covering a wide range of real scenarios. Yasushi Sumi and Fumiaki Tomita in their work [75] developed a new segment based technique for object recognition based in 3D models information extracted by stereo vision. Li Jingchao et al. in [39] proposes a 3D reconstruction approach based in stereo vision and texture mapping, referring that it could be used in a vast field of areas like visual navigation of robots to 3D games, digital library, visual communication, virtual reality, internet travelling, etc.

An interesting work is the one present in [2], where the authors main focus is to replicate the data returned by a Laser Range Finder (LRF) (sensor presented further in this Chapter) using a Stereo Vision system. The idea is that many robotic solutions (more in mobile robotics) base their approaches in a combination of LRFs and cameras to simultaneously acquire some visual information and the depth map data. In this sense, replacing the usage of two sensors by a stereo vision system, presents a great advantage in terms of system reduced complexity and lower costs.

The problems of using vision based system in a manufacturing environment are significant. For example, environment light changes and random noise that influence the performance of solutions based on CCD cameras.

3.1.4 Time-of-Fight (ToF) and Laser Range Finder (LRF)

To obtain the 3D model of a surface Time-of-Fight (ToF) sensors can also be used (see Figure 3.4). Their working principal is based on the emission of laser pulses which reflects in the target object.

Figure 3.4: Time-of-Flight sensors.

The receiver detects the reflected pulse measuring their intensity and travel time (see Figure 3.5). Considering ToF, Yan CuiSchuon et. al [13] describe a method for 3D object scanning by aligning depth scans that were taken from around an object. The authors refer that their approach overcomes the sensor's random noise and the presence of a non-trivial systematic bias, by showing good quality 3D models with a sensor with such low quality data, see Figure 3.6.

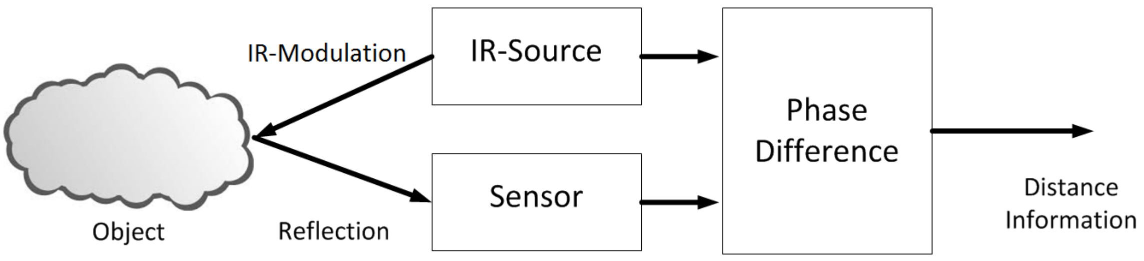


Figure 3.5: Time-of-Flight working principal.

Due to comparably simple technology that these sensors have, the authors see a bear potential for low cost production in significant volumes.

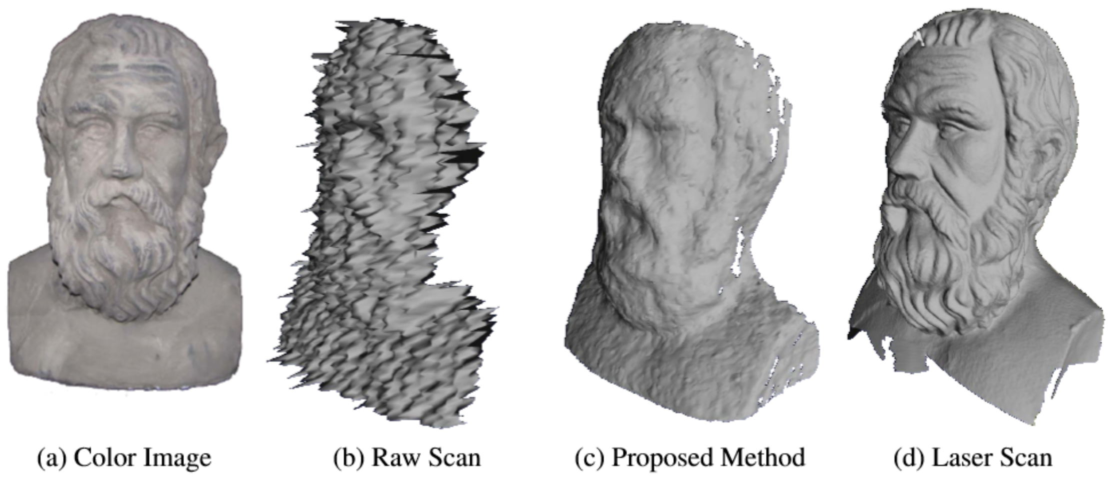


Figure 3.6: TOF modelling results with the method proposed by Yan CuiSchuon et. al [13].

S. Schuon et al. in [71] shows that ideas from traditional colour images super resolution can be applied to ToF cameras in order to obtain good 3D data information. These authors also make reference that the depth measurement of ToF sensors suffer from random noise and systematic error.

Although a good effort has been made, the quality of the 3D modulation based in these sensors is lower when compared with the alternatives already presented.

Considering now Laser Range Finder (LRF) (Figure 3.7) their working principal is very similar to he ToF sensor - send a laser pulse in a narrow beam towards the object and measure the time taken to the pulse to be reflected. In the present days 2D and 3D laser ranger finder are available.

Figure 3.7: Hokuyo and Sick Laser Range Finders.

Using LRF [7, 31, 78] present a research work where this sensor is used to perform 3D scene reconstruction. However, this is usually done in mobile navigation and not with industrial systems (precision requirements). The disadvantages of LRF are their high price for high precision measurements and the measurement variation with the object reflective properties. Although laser based solutions are the most used in industrial environment, usually the choice falls on laser beam sensors.

In more recent works, [27] and [32], it is presented a new approach for 3D Modelling that allows the extraction of 3D shape and colour. This system is based on a two-dimensional LRF and a camera, interesting if colour feature is important for object distinguishing.

In the end, and for the present research work, the solution for 3D modelling has to consider the unique characteristics that may distinguish each of the objects/products like colour, texture or/and shape. The robots environment limitations where the recognition system will be mounted also needs to be considered.

Note that, although the final objective is to integrate all the work with an industrial partner, some preliminary tests using 3D modelling sensors were performed at a laboratory framework.

From the state-of-art solutions, and considering precision requirements, CCD Camera, LRF and Laser triangulators were the selected approaches to be explored. For each test, different objects were considered that will be presented to the reader during the Chapter's development.

3.2 Laser Range Finder vs. CCD Camera

Almost since the beginning, computer vision has been used on industrial environments, allowing robots to perform important tasks like quality control, inspection and recognition. Vision systems are typically used to determine the position and orientation of objects in the workstation, enabling them to be transported and assembled by a robotic cell (e.g. industrial manipulator). These systems commonly use CCD (Charge-Coupled Device), and CMOS (Complementary Metal-Oxide-Semiconductor) Cameras fixed and located in a particular work area or attached directly to the robotic arm (eye-in-hand vision system). Although it is a valid approach, industrial environment is very harsh and aggressive, where artificial vision systems based on ordinary cameras tend to be influenced by magnetic interference and variations in ambient light [26]. Considering this scenario, a novel approach is emerging where laser approaches start to dominate the field.

In this section a comparison between camera, LRF and laser triangulators will be performed.

Starting by the LRF, and for the results presented in this section, the 2D LRF will be attached to a robotic manipulator (see 3.8), which will execute a pre-defined path to produce range images of the scene. With this technique, the environment light interference is minimized resulting in a more reliable and robust computer vision system.

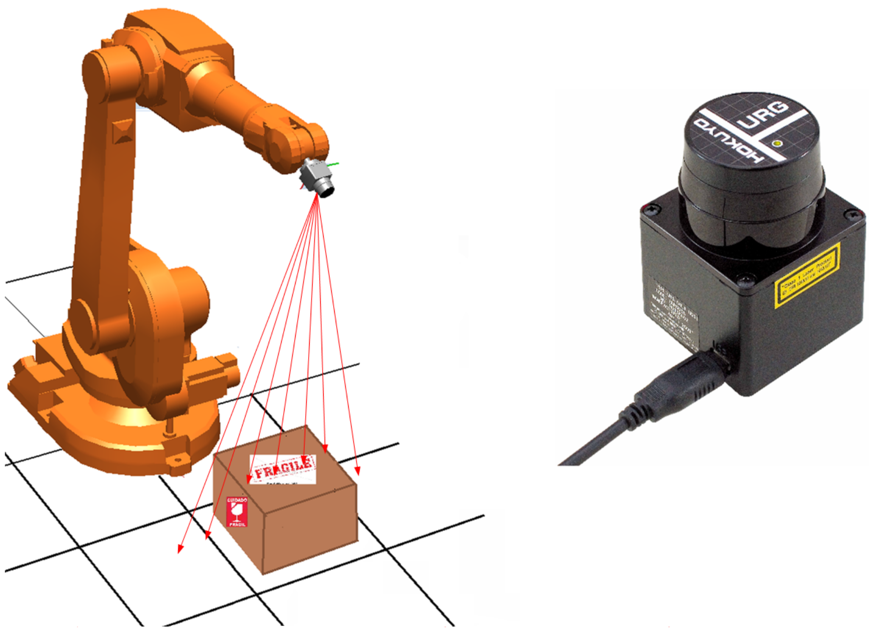


Figure 3.8: Manipulators workspace scanning concept and Laser Range Finder used.

To obtain these images, the manipulator only needs to scan the workplace where at each time the distance between the laser and the object present in the workstation is indicated. Subsequently, the image is scaled in shades of grey (dark grey indicate smaller distances, and vice versa for light grey), and a 2D representation of the scene is obtained, where every pixel has information on depth.

LRF are sensors often used in mobile robotic applications. These sensors are equipped with a mechanical system that makes a mirror spin with an angular resolution, allowing the IR light to cover in sequence a set of straight segments with different orientations. Because of the mechanical associated constraints most of the sensors performs a series of measures within a limited angle, below 360 degrees, which originate an area called blind or dead zone.

In this work it was used a Hokuyo Automatic Laser Range Finder, URG-04lx [3], which uses the difference of phase between the emitted and received infra-red rays, with a wavelength of 785 nm to estimate the distances to the objects. This LRF is a security Class 1, with a scanning area of 240 degrees, angular resolution of 0.35 degrees per step and performs a complete full scanning in 100 ms. In a single scan, it can take about 683 steps (measures), and this information is transmitted by RS-232C or USB. It has a range between 60 mm and 4095 mm and an error of 10 mm for measures lower than 1m, and 1% for measures higher than 1 m.

3.2.1 Concept of Exploratory Scan

To understand the algorithm developed for the image creation, it is fundamental to explain what is the meaning of exploratory movement.

Based on the assumption that a 2D LRF is used, coupled to the robotic arm, at each time a plane of distances that could be seen as an array, is projected. Therefore, to sense the workspace, it is necessary to produce some laser movement around the scene. This movement was designated as exploratory movement and aims to reach and characterize all desirable points (table, ground, parts, etc.). In the considered set-up, the LRF moves parallel and horizontally to the workstation, like illustrated in Figure 3.8. During the exploratory movement, the laser provides a continuous range of distances (each distance D_i has an angle θ_i) associated. Upon receiving the scan, the distances are converted to XY points (relatively to the laser reference). The X-axis reflect the vertical distance and the Y-axis constitute the horizontal distance of the measured point (the relative separation).

In a simple matter, the X-value of each point (D_X) provides information on their depth, scaled in values between 0-255 using the height of the laser to the workstation. The Y-value (D_Y) offers information on the separation of some points with respect to the centre j of the laser scan, where $\theta_j = 0$ (Figure 3.9). Considering that, an array with dimension of $1 \times ImageWidth$ was created, the index is obtained by the Y-value.

Obviously not all the points given by the laser are used to create the image (see Figure 3.10). Instead, only those that are within an angle range of $[-\Delta\theta;\Delta\theta]$ are used. This

[3]http://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx_ug01.html

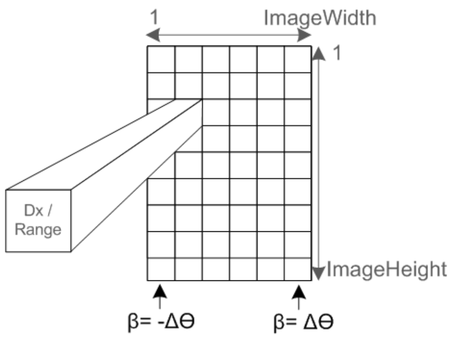


Figure 3.9: Structure of the image created by the scanning system.

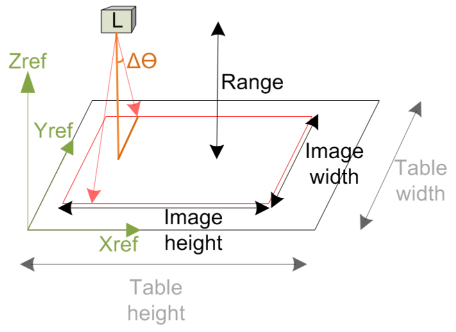


Figure 3.10: Using the laser distances to characterize the environment.

interval is automatically determined from the height of the laser to the work plan, the width of the desired image and the image resolution. After this point, the X-value and Y-value (vertical and horizontal distance - D_X e D_Y) are computed.

3.2.2 Approach Results

To test both vision systems (CCD Camera and LRF) 7 pre-defined objects were considered: turn screw (Figure 3.11a), industrial spray-paint (Figure 3.11b), industrial painting tank (Figure 3.11c), small water bottle (Figure 3.11d), artificial wood bar (Figure 3.11e), wood block (Figure 3.11f) and cork block (Figure 3.11g). In the next two sections, the results for each of the 3D modelling approaches are presented.

3.2.2.1 Laser Range Finder (LRF)

Figure 3.11 shows the application of the LRF approach to each of 7 different objects. Each observation sample consists of two images: one reflects the result of the laser scanning and the other reflects the result of binarization and boundary extractions.

As it is possible to observe well defined boundaries between the environment and the object of interest were computed (considering the simple scenario mounted), resulting in

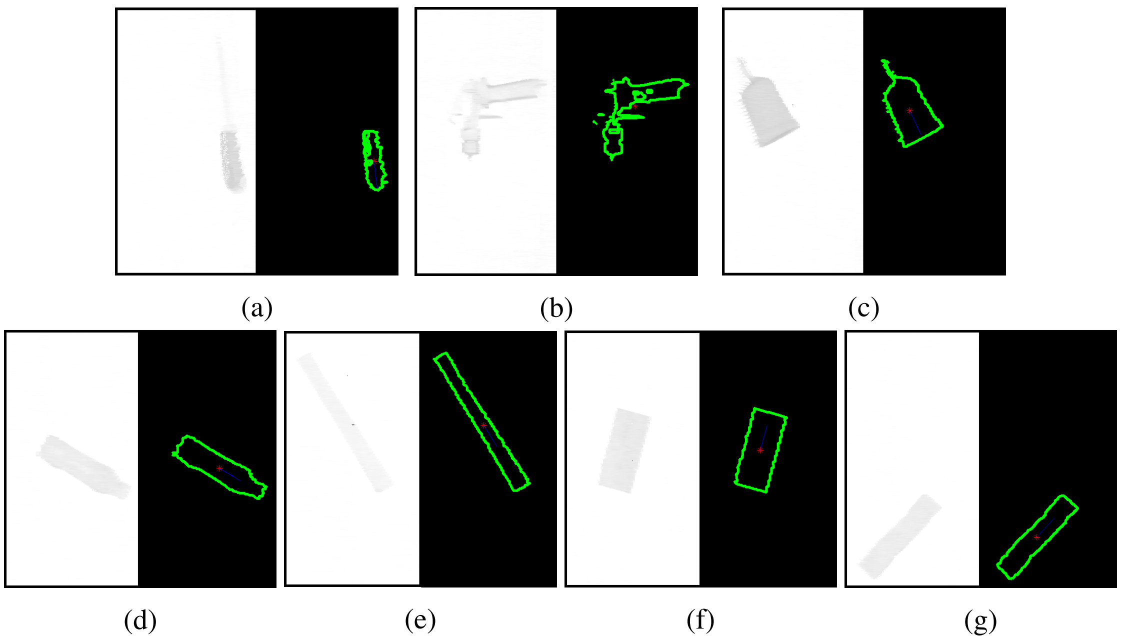


Figure 3.11: Samples of objects obtained by the LRF. On the left side of each sample are the original LRF image. In the right side the Otsu's threshold method is applied plus perimeter delimitation using connectivity 8 (Matlab function pwperim).

a minimization of the features variance, and thus allowing a more accurate classification. Note that, any special environment light conditioning was performed.

It is important to mention that all images were taken setting the industrial robot speed equal to 600 mm/min and with a perpendicular distance of 58 cm from the platform where each object was placed.

3.2.2.2 CCD Camera

To prove the higher robustness to light variations of LRF over the CCD camera, experimental tests using an RGB Camera and the same simplistic scenario were performed. The CCD Camera will be fixed above the parts plane of interest.

The images obtained are presented in Figure 3.12. The same thresholding method algorithm was used both for range images and intensity images, to faithfully compare and interpret the results.

Figure 3.12 shows the original RGB camera images and the corresponding Otsu's segmentation results. This step, segmentation, is the most sensitive to environment light changes. From these images, the Hu's feature extraction and classification processes are similarly applied to both LRF and CCD Cameras approaches.

Figure 3.12a illustrates an ideal case, where high contrast between the work object and the background is present. Therefore, the application of the Otsu's segmentation method

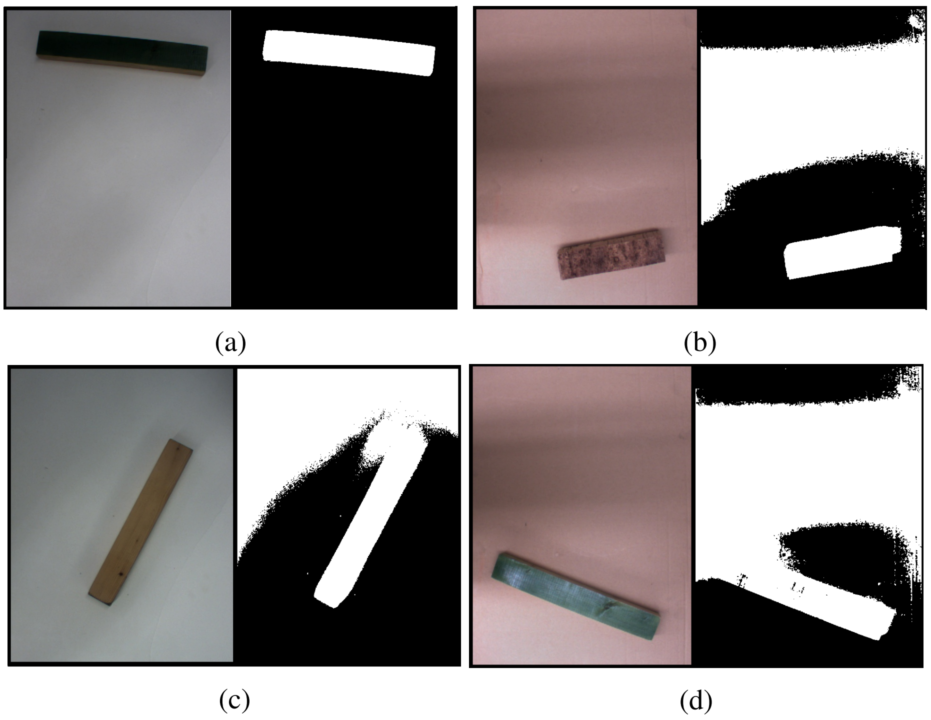


Figure 3.12: Samples of objects obtained by the CCD Camera approach. In the left side of each sample the original camera image is show. In the right side it is presented the Otsu's threshold method.

is performed with a successful result, however for the remaining images this segmentation was unsuccessful which forbids the feature extraction process. Figures 3.12b and 3.12c represent the low contrast problem between the work object and the background, and finally, the Figure 3.12d illustrates the environment's light problem. Although it was possible to include different image processing techniques to remove object background from the images, it stays the fact that CCD cameras are unreliable in an unstructured light environment and with a low contrast between object and background colours cases. Although not impossible the extraction of the objects of the scene , the task increases in difficulty. Furthermore, for industrial purposes a robust and reliable sensor must be used, where LRF gives more guarantees.

3.3 Laser Range Finder vs. Laser Triangulators

During the research development, the industrial partner (that will be presented in Chapter 6) provided some of their products. Therefore, and considering that industrial applications are the core of the present work, it makes sense to perform the following tests in these new parts (see Figure 3.13).

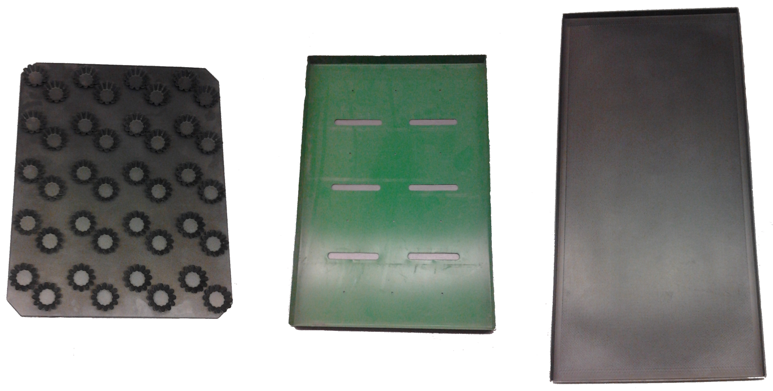


Figure 3.13: Example of some work parts available - Object Dimensions [400-600] width x [650-800] mm height.

This new scenario will allow to test immunity of LRF to other properties like parts colour and reflection properties.

3.3.1 Experimental Set-Up at the Laboratory

Figure 3.14 presents the laboratory set-up built to test both sensors. As it is possible to see, the Laser beam sensor was placed in a central position relatively to the part. For this vision system, the movement necessary for the 3D model extraction is performed by a conveyor. The set-up considered is also adequate for the LRF approach.

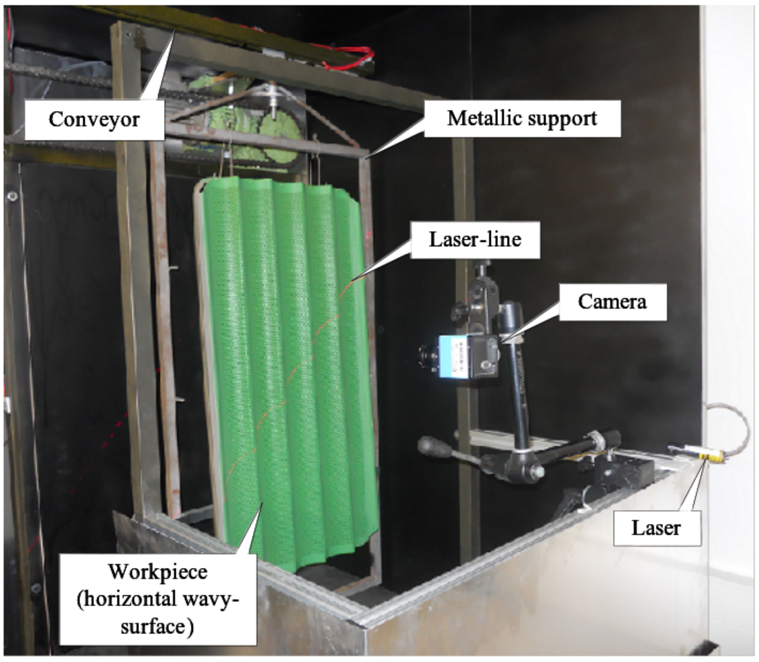


Figure 3.14: Camera-Laser laboratory test set-up (Part Size in the figure 800 x 300 mm).

Note that, measures were taken so the support carrying the part cannot suffer oscillations in the axis normal to the conveyors movement, since it could hinder the 3D modulation of the objects of interest.

3.3.2 Approach Results

With the described set-up, several tests were performed for each of these two approaches.

3.3.2.1 Laser Range Finder

Starting by LRF, the same approach presented earlier but now in a vertical position was applied to these new parts and some problems emerged (see Figure 3.15).

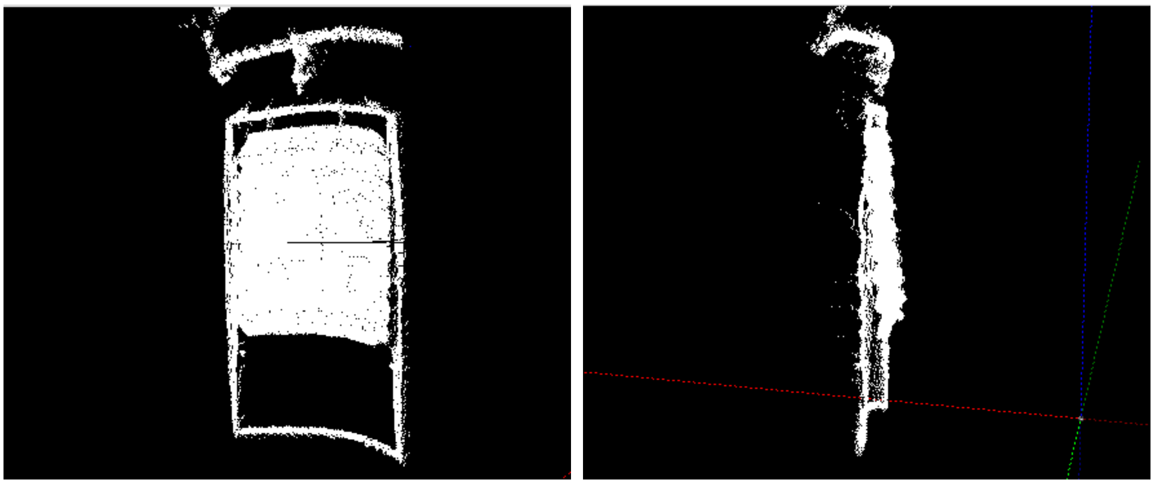

(a) (b)

Figure 3.15: Part Type modulation example with Hokuyo's Laser Range Finder.

As it is possible to observe in Figure 3.15, the models present a high value of noise. Besides, it has also verified a deformation of the 3D model mainly related with the incident angle of the LRF in the object.

Beyond the already presented problems, the measurement variation of the LRF with the colour properties of the object was also studied. In this way, and in Figure 3.16, it was placed a 2D chessboard in front of one of the new objects.

From Figure 3.16, it possible to see that the black and white squares of the chessboard were modelled differently. These issues are related with the different absorption properties (well known) of each colour. This feature is not desirable for object recognition, especially in cases where the object can change its particular colour during the production process and where the recognition procedure needs to be performed more than once.

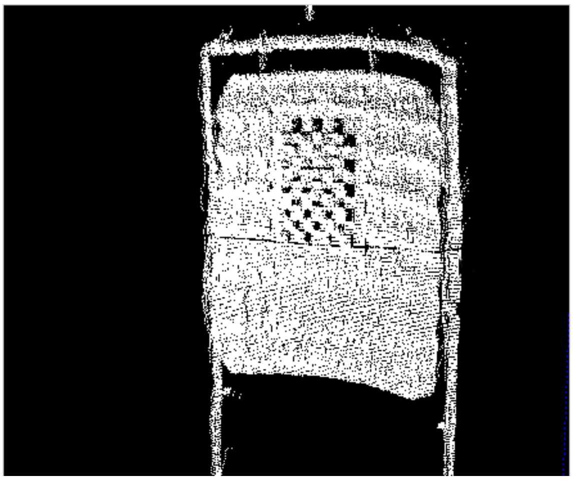


Figure 3.16: Part Type modulation example with Hokuyo's Laser Range Finder - Inserted a 2D chessboard.

Note that, for this work, the focus is to develop a low cost system that small enterprises can have access, and where accuracy in classification is one of the most important prerequisites.

In Appendix A is presented a simple description of the software developed for the extraction of the objects 3D model using the LRF sensor.

3.3.2.2 Structured Light Sensor

As explained earlier, the results based on a low cost LRF were not satisfactory. Hence, the approach using Camera-Laser triangulation system for the construction of the parts 3D model was explored. Measures were taken to have a structured light environment crucial for image capture when using CCD cameras (the disadvantage of the approach).

In the proposed set-up, and as shown in figure 3.14, the laser and the CCD Camera (Characteristics: grey image and resolution: 1024x768) are located in a central position of the part. The part is then fixed through a support attached to the conveyor, allowing the production of the required motion for CCD Camera and laser beam triangulation system to perform 3D model extraction.

Camera Calibration The first step was directed to the system calibration. All the related procedure need to be easy so any operator without calibration knowledge can perform it. Therefore, the developed approach was based on one dot board with known dimensions, which allows the computation of the intrinsic and extrinsic parameters of the camera using Tsai camera calibration method [76] and [38].

- Extrinsic Parameters - provide information on the position and orientation of the camera relative to some global coordinate system (or the world).

- Intrinsic Parameters - provide camera internal optical and geometrical characteristics (focal length, scale factors, position in pixels of the orthogonal projection of the optical centre in the plane of projection and distortions offered by lenses).

The origin of the world coordinate system was defined as the centre of one of the dots in the calibration board.

Laser Plane Calibration Having computed the intrinsic and extrinsic parameters of the camera, the computation of the laser equation plane was trivial to obtain. For that, a world coordinate reference laser point was measured (its measuring precision is important to not introduce errors in the laser calibration). Then and by forcing the laser beam to cross the origin dot in the calibration board it is possible to compute the 2D vectorial line equation (the z coordinate is zero since the laser line is projected in the dot board). From the 2D line vectorial equation and laser point in the world's reference frame, the laser plane equation is computed. In summary, the laser equation plane (L), the pose of the camera (T^w position) and its intrinsic values were computed.

Now to perform the triangulation it is only necessary to compute the intersection point (p_x^w, p_y^w, p_z^w) between the laser plane with the line r, see figure 3.17.

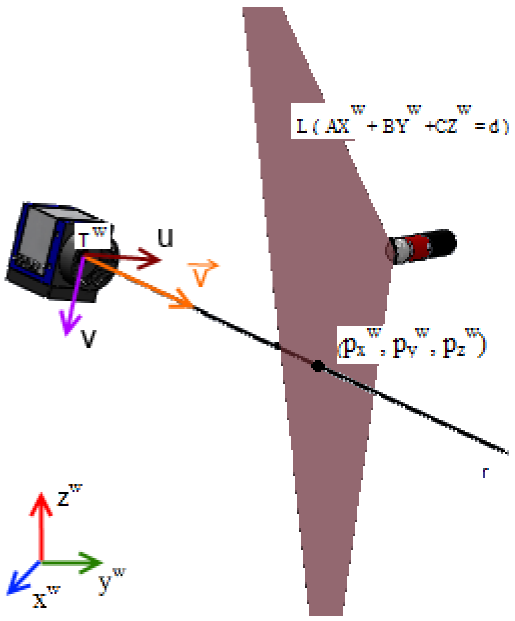


Figure 3.17: Laser Plane and CCD Camera interception point.

Therefore, having the function that converts a pixel (u,v) to a 2D point in the world (x^w, y^w) the line r directional vector $(\vec{v})$ can be computed by:

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} p_x^w - T_x^w \\ p_y^w - T_y^w \\ 0 - T_z^w \end{bmatrix} \tag{3.1}$$

Where $[T_x^w, T_y^w, T_z^W]$ is the position of the camera in the world reference frame and $[x^w, y^W, 0]$ is a laser point in the camera view. Therefore, the line r vectorial equation is equal to:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + k \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \tag{3.2}$$

After computed $[V_x, V_y, V_z]$, p^w can be calculated by solving the next system of equations:

$$\begin{aligned} p_x^w &= T_x + KV_x \\ P_y^w &= T_y + KV_y \\ P_z^w &= T_z + KV_z \\ Ap_x^w + Bp_y^w + CP_z^w &= d \end{aligned} \tag{3.3}$$

Figure 3.18 shows the modulation of some of the new objects considered, with all the system calibrated.

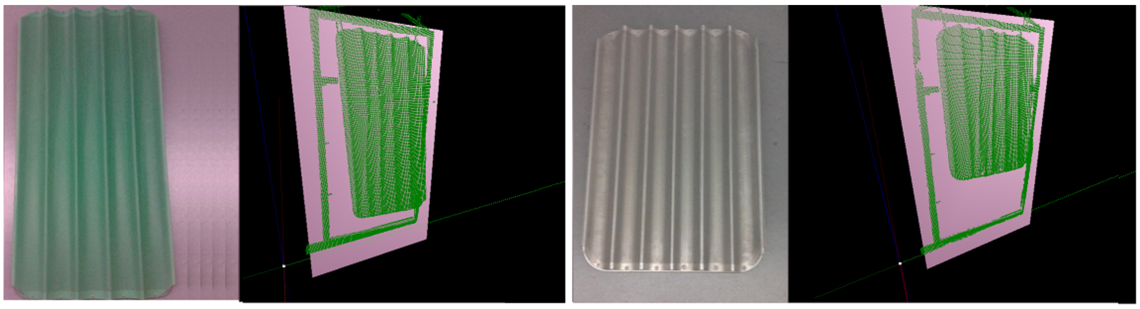


Figure 3.18: 3D Modulation Examples performed by Camera-Laser Triangulation system.

On the left side of the images is presented the real object, and in the right the model obtained by the Camera-Laser triangulation system.

As it is possible to see, models with low noise level when compared with LRF were obtained. This solution is much more robust to object reflection properties and colour, however, structured light environment is necessary.

Beyond the problem of light requisite, Camera-Laser triangulation sensor has other well-known limitations:

- Occlusion: Due to some object shapes, the laser light source, can sometimes be hidden from the camera view. This will cause the loss of object information. However different approaches can be followed to avoid these limitations, as such, the use of a second CCD camera [5, 36, 81, 11], the use of a second laser line [37], or laser higher triangulation angle [12, 18]. The two starting solutions can be used to increased at the same time the precision of the system.

- Reflection: when the object has reflective properties it is possible that the laser beam that focuses on the objects surface, to be reflected elsewhere. This phenomenon causes serious problems, since the reflective part is not in the laser plane, resulting in high measurements errors. One solution it to tune dynamically the parameters of the camera and delete unrealistic data.

3.4 Conclusions

The conclusions that are possible to extract from these tests are that: despite the high immunity level of LRF to environment light, it presents problems related with noise and reflection properties related with objects material and colour. Note that, in the presented experiments a low cost LRF was used. Due to these disadvantages, the approach was eliminated from the possible ones.

Considering the CCD cameras, and analysing their susceptibility to the environment light when compared to LRF, it turns the choice easy to be made. Moreover, for the reconstruction of 3D objects models a moving camera or two fixed cameras needed to be used, turning the system more complex than the required.

Considering now the solution based in structured light (CCD Camera plus Laser Beam), it was possible to verify the increase of model construction reliability, due to its immunity to colour, noise measurements and in a certain manner more robust to reflection than the LRF. Note that, like the name implies structured light environment is needed, due to the use of CCD Cameras.

In this sense, it is not possible to say that a specific solution is better that the other one. This choice is directly related to the application, environments and object constraints.

Therefore, and considering objects constraints minimization, laser triangulation approach was the selected solution.

Chapter 4

Experimental Results in Learning Approaches

Having presented earlier the architecture for object recognition, a comparison between three different algorithms (that were discussed in Chapter 2) k-nearest neighbour, Neural Network and Support Vector Machine will be done in this Chapter.

To perform this comparison, seven 3D object models were extracted (turn screw, industrial spray-paint, industrial painting tank, small water bottle, artificial wood bar, wood block and cork block) with the laser Range Finder (see Figure 3.11).

Furthermore, for the implementation of the Machine Learning methods all the architecture described in Chapter 2 - Figure 2.15 was used. As explained, in this architecture the simulated annealing selects an initial feature vector, and then the K-fold cross-validation algorithm adjusts the best model parameters for the classifier. At the end of K-fold cross-validation, the best classifier parameters and the respective accuracy return back to the feature selection phase, and the process is repeated until the stopping conditions (the temperature concept inherent to the algorithm or the maximum iteration number) are reached. From this, the best combination of features/model characteristic (corresponding to the best training accuracy value) are saved. The generalization error can be predicted in the final phase of the algorithm, by measuring the performance of the estimated model in independent data (test data). Due to the iterative heuristics, K-fold cross-validation algorithm and simulated annealing, the computation of the best feature subset and the classifiers models parameter tuning takes some time to perform (dependent of the number of classes to train, the number of simulated annealing iterations and the number of folds considered in K-fold cross-validation procedure). Nevertheless, this procedure is performed off-line.

The consideration of the three machine learning algorithms (kNN, NN and SVM) will also allow the comparison of the balance between the complexity and expected accuracy of each type of classifier model. To point out again that all the recognition system is independent of the type of sensor used for modelling purpose.

4.1 7 Objects - Feature Analysis

As previously referred, and having already extracted the object 2D image, the first step is the object's feature extraction procedure. For the tests performed the features considered were the Hu Moments (I1 to I8 in Table 4.1), Area, Perimeter and Eccentricity.

To demonstrate the features extraction algorithm, some examples of objects are show in order to prove that the desirable features represent well the object characteristics . Each sample (observation) consists of two images; one reflects the result of the laser scanning (converted to grayscale) and the other the result of feature extractions.

4.1.1 Same Object - Feature Analysis

First are compared the features of the same object with different position and orientation, see Figure 4.1 and Table 4.1.

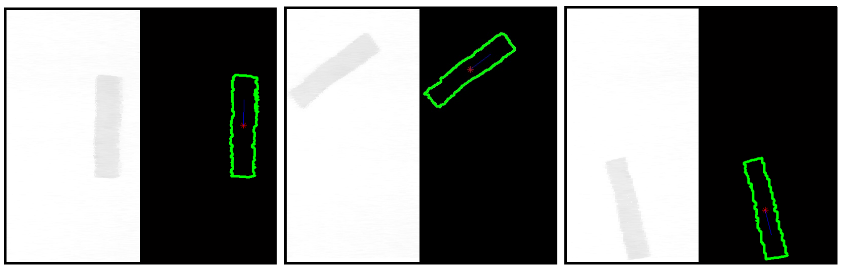


Figure 4.1: Samples of tree different scene observations for the same object.

Features Extraction:										
I1	I2	I3	I4	I5	I6	I7	I8	Area	Perim.	Eccent.
0,435	0,161	5,04e-08	3,34e-08	1,03e-15	1,32e-08	-6,56e-16	-1,06e-09	7914	611	4,857
0,407	0,137	2,33e-08	5,20e-09	1,18e-17	7,52e-10	3,58e-17	8,90e-10	8057	649	4,503
0,371	0,110	2,67e-08	1,77e-08	3,54e-16	5,65e-09	-7,49e-17	8,06e-10	9388	683	4,087
Standard Deviation										
0,03	0,03	1,47-08	1,41E-08	5,18-16	6,27E-09	3,71E-16	1,10E-09	812,88	36,01	0,39

Table 4.1: Features Extraction - different observations for the same object.

As it is shown in table 4.1 some features for the same object are relatively constant, like the Perimeter, Eccentricity and I1. These features may give good indications for object specific characterization.

4.1.2 7 Different Objects - Feature Analysis

Now, and considering the seven different objects to be distinguished, Figure 4.2 and 4.3, the same feature analysis is performed. The results are presented in Table 4.2.

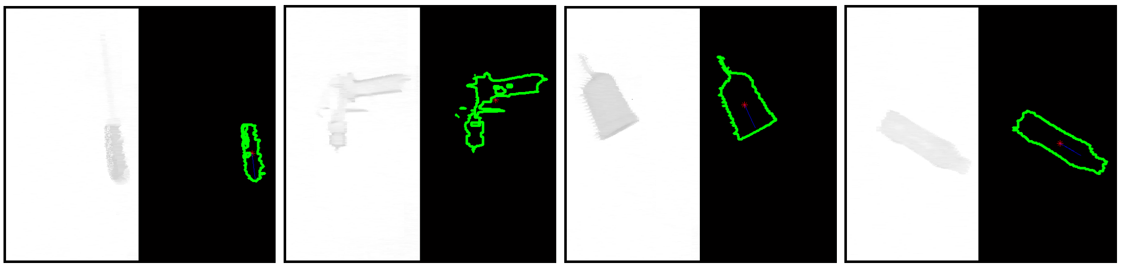


Figure 4.2: Sample of a turnscrew, spray-paint, painting-tank and water bottle in the robotic environment.

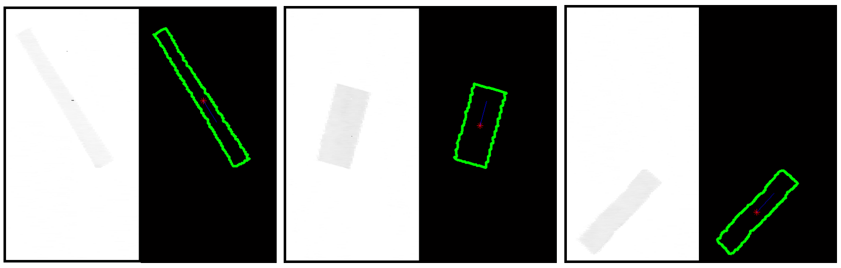


Figure 4.3: Sample of a artificial wood bar, wood block and cork block in the robotic environment.

For the selection of the objects used for the experiments, their differentiation properties were considered . In this way some objects are easy to identify, but there are others that their classification is not so trivial, like the distinction of the artificial wood bar, wood block and the cork block.

Features Extraction:										
I1	I2	I3	I4	I5	I6	I7	I8	Area	Perim.	Eccent.
0,907	0,794	1,54e-06	1,37e-06	3,16e-13	1,22e-06	3,58e-13	4,34e-08	8936	947	10,39
0,397	0,130	6,70e-09	5,96e-09	4,96e-18	2,14e-09	-2,36e-17	4,87e-11	8216	650	4,409
0,230	0,025	6,74e-11	7,58e-11	3,18e-21	1,211e-11	2,23e-21	2,11e-14	10170	563	2,119
0,341	0,089	4,32e-08	2,27e-08	4,85e-16	6,71e-09	-2,13e-15	5,11e-10	8585	659	3,728
0,202	0,011	1,11e-07	1,91e-08	1,73e-16	1,97e-09	-4,68e-16	-2,31e-10	8447	657	1,483
0,507	0,110	1,13e-05	1,13e-06	-3,90e-13	-2,54e-07	2,79e-12	1,32e-07	7007	1064	1,949
0,335	0,080	1,66e-07	1,12e-07	1,01e-14	3,11e-08	-6,15e-15	3,36e-09	3018	571	3,324

Table 4.2: Features Extraction - Example for all the objects considered.

By analysing table 4.2, it is possible to conclude that eccentricity, Area, Perimeter and I1 and I2 will be good features to be considered for object distinguishing [1].

In the following sections, the results of applying the classification architecture to different types of classifiers will be presented. As explained before KNN, NN and SVM will be used, and for each one it will be analysed the subsets of features that most contribute to the correct classification. It is important to note that the feature subset vector, in the training phase, is a binary array where each "1" represents the activation of the correspondent feature and the "0" is a not-active feature.

The results for each experiment present the best feature vectors width lower training error (training accuracy) and the respective tuning parameters of each classifier. Finally, the generalization error for each estimated classifier model is presented. This generalization error is computed by applying the model estimated and tuned to an independent data set (test data).

4.2 Comparison Between SVM, kNN, NN

Before presenting the comparison results, it is important to refer that, for each object it was extracted 33 samples (value tunned for training and testing after the performance of some initial tests). Therefore, with the selection of 7 different objects the data set will be constituted by 231 samples. From this set, 20 samples of each object (balanced class data set) were used for training and the remaining for testing purposes. This separation was made since it was considered that about two thirds of the data for training purposes and one third to testing purposes. These values are normally used by state-of-the-art procedures for short datasets.

Considering the simulated annealing parameters, the temperature was set to be equal to 200 (temperature decreasing in each iteration), a *racioTemp* of 0.95 and the iteration number defined as 25. These parameters were selected after the performance of initial tests.

4.2.1 k-Nearest Neighbour (kNN)

The first approach tested was kNN. Knowing the k closest neighbours and their belonging classes (train data), it was used the mode function of the neighbours to select the class for the test sample.

For the metric used to obtain the kNN, Euclidean distance was selected.

[1] Although a "visual" feature analysis was made, the selection of the best features will be performed autonomously by the simulated annealing procedure.

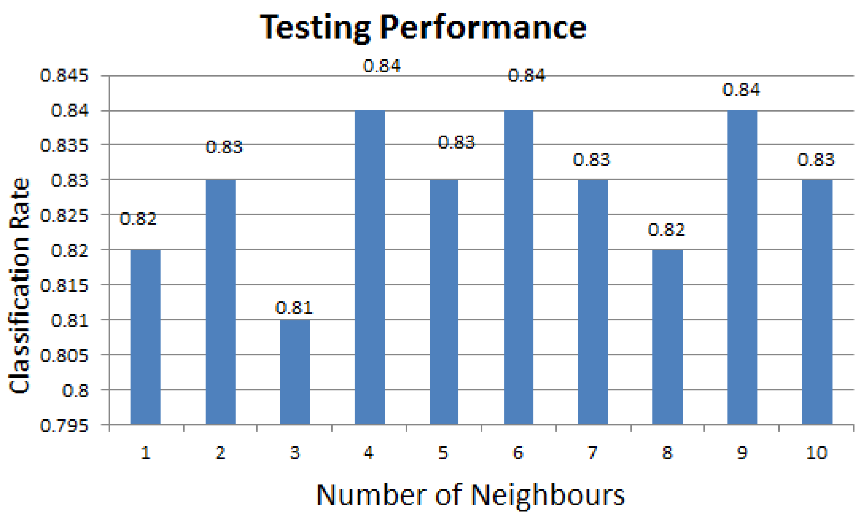


Figure 4.4: Result of the Application of k-NN algorithm.

In figure 4.4, it is presented the accuracy (generalization error) obtained for the different values of neighbours considered. By analysing these results, it is possible to verify that the generalization error is not very sensible with the changes performed in the number of neighbours value (k). This can induce the error to say that the data is well separated into classes. However, only a classification rate of 84.0% was achieved. This may indicate that there exists noisy data in the train/test set or there is some non-linearity in classes that this method could not capture.

These accuracy values were obtained for the features selected using the simulated annealing process introduced before, i.e., after the generation of the first solution using a predefined feature combination, the algorithm will search other possible combinations that possibly will increase the value of the objective function (accuracy). In these tests, K-fold cross-validation was not used for parameter tuning, since the only variable in here presented is the number of neighbours, and a study of the variation of the generalisation error with the number of neighbours is interesting to be analysed.

K Value	Features Selection:											No. of Features:
	I1	I2	I3	I4	I5	I6	I7	I8	Area	Perimeter	Eccentricity	
3	1	1	1	0	1	0	0	1	0	0	1	5
8	1	1	0	0	0	0	0	0	0	0	1	3

Table 4.3: Best Features Combination.

In table 4.3 are present the features combination that retrieved the maximum classification rate for *k* equal to 3 and 8, used to classify the test sample according to its neighbours. Features I1, I2 and eccentricity are presented in most of the results, meaning that they have a bigger importance to the kNN classification algorithm. Comparing these features selected by simulated annealing with the ones selected visually in the beginning of the Chapter, it is possible to conclude that simulated annealing retrieved the expected results. This algorithm will be useful in cases where the classes unique features are not so trivial to visually analyse or in cases where there is not only a specific feature that separate the object but a combination of several ones.

4.2.2 Neural Networks (NN)

NN are an algorithm much more complex than kNN, with much more parameters to tune (example: neurons activation function, number of hidden layers, number of neurons for each hidden layer, etc). Therefore, it was necessary to fix some parameters, which are: activation function of the hidden layers (sigmoid function the most used one) and the activation function of the output layer (softmax). The softmax function was used since an output layer with 7 neurons which correspond to the 7 classes was built.

Softmax function deals with the classification in a probabilistic way giving in the output layer the probability of the data to be from the different classes. In an ideal situation, only one neuron in the output layer should have a value different than zero, whose index will correspond to the object class.

In the approach developed the number of hidden layer was varied between $\{1;2\}$ and the number of neurons in each layer, between $\{2;5\}$ in the first hidden layer and $\{2;3\}$ in the second hidden layer. These values were considered since a NN with one hidden layer can represent any continuous function and with at least two hidden layers for discontinuous ones. The number of neurons in each layer was chosen after the performance of some initial tests.

The number of neurons in each layer was tuned using the K-fold cross-validation cycle. In this way, K-fold cross-validation is used to try to obtain a more reliable model performance measurement, and at the same time tune the parameter presented before.

Next are presented the results obtained for one and two hidden layer respectively.

4.2.2.1 Neural Network - Fixed with one hidden layer

For one hidden layer, the best result was obtained for 4 neurons in the hidden layer and feature combination presented in table 4.4, with a training accuracy of 94%. The accuracy value resulting of the application of these tuned parameters in the test set, was equal to 95% (generalization error).

NHL	Features Selection:											No. of Features:
	I1	I2	I3	I4	I5	I6	I7	I8	Area	Perimeter	Eccentricity	
1	0	0	0	1	0	0	0	0	1	0	1	3

Table 4.4: Best Features Combination for one hidden layer (NHL).

4.2.2.2 Neural Network - Fixed with two hidden layers

Considering now two hidden layers the best result was obtained for 2 neurons in the first hidden layer and 3 neurons in the second hidden layer, and feature combination presented in table 4.5, with a training accuracy value of 96%.

NHL	Features Selection:											No. of Features:
	I1	I2	I3	I4	I5	I6	I7	I8	Area	Perimeter	Eccentricity	
2	0	0	0	0	0	0	0	0	1	0	1	2

Table 4.5: Best Features Combination for two hidden layer (NHL).

The generalisation error, obtained for the application with these parameters in the test set, was equal to 95.5.

4.2.3 Support Vector Machine (SVM)

Finally to consider SVM classifier. SVM is a useful tool for classification purposes. However, it is a binary classification algorithm, and for problems with several classes two main SVM based approaches are normally used: One vs one and One vs all. The literature is inconclusive about the best approach to solve multi-class problems, therefore, a study comparing the performance achieved by each approach was performed.

Next are presented the results of applying the algorithm explained in Figure 2.15, and based on that, some parameters can be analysed: the feature subset, the C-value (soft margin parameter) $\in [0.25, 4]$ tuned by K-fold cross-validation, and the generalization error (the expected real accuracy of the model). The defined interval of possible values for the C parameter, was tuned after the performance of some random initial tests and evaluating the generalization error returned from the SVM tuned model.

Other important parameter for SVM models is the kernel function. For the research work it was used both Polynomial, $K(X_i, X_j) = (X_i.X_j)^D$, and the Radial Basis Function (RBF), $K(X_i, X_j) = exp\left(-\gamma \|X_i - X_j\|^2\right)$, where the different parameters were selected (polynomial order, and the gamma for the RBF) according to the best performance achieved on the training phase.

For each result, it was saved the feature subset, C-value, the kernel parameter, and the training accuracy of the model.

In the following graphs are presented the training performance, that is an optimistic accuracy, and the generalization error computed by applying the SVM model to an independent test set.

4.2.3.1 Radial Basis Function

For the Radial Basis kernel function, the gamma value for the RBF was defined as $\gamma = i/(SizeTrainingData)$, where i (the gamma factor) was changed according to the following graphics.

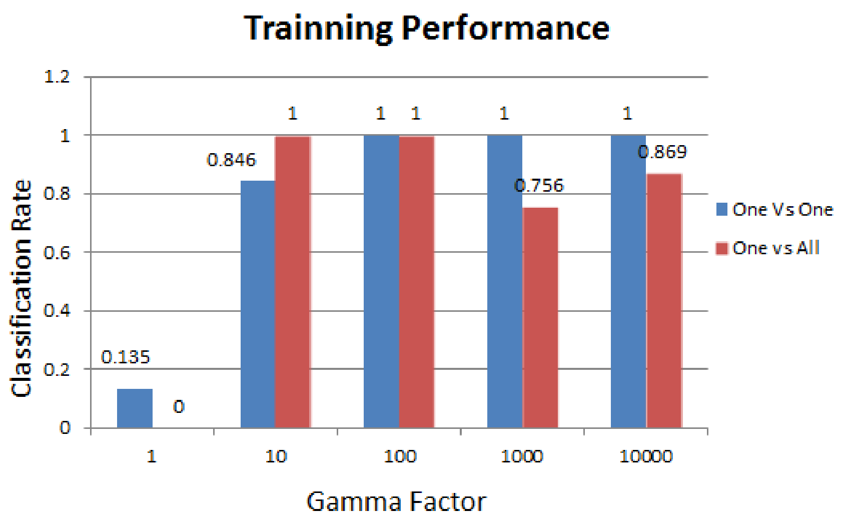


Figure 4.5: Training performance obtained as a function of gamma factor.

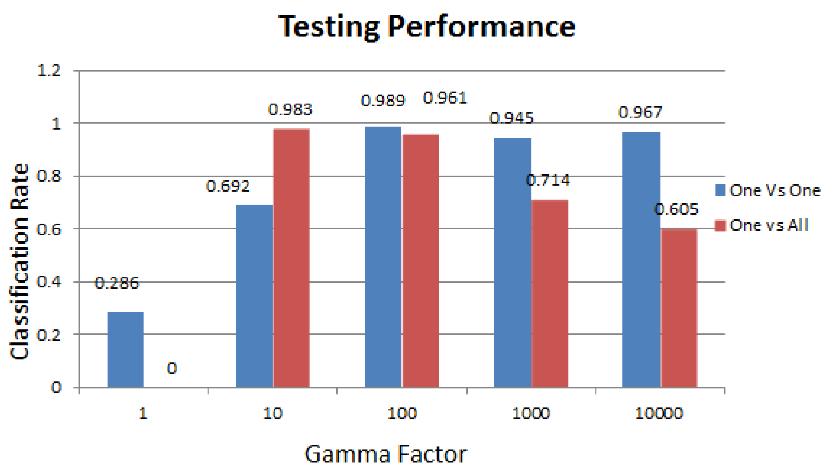


Figure 4.6: Testing performance obtained as a function of gamma factor.

In the graphics 4.5 and 4.6 it is possible to see, that different performances (training and generalization error) were achieved when is used the strategies One Vs One, and One Vs All. In a general terms, the One Vs One strategy obtained better results, specially when $i = 100$ where the generalization error has a value of 98.9%. For that accuracy the features selected are presented in table 4.6.

Features Selection:											No. of Features:
I1	I2	I3	I4	I5	I6	I7	I8	Area	Perimeter	Eccentricity	
0	0	0	0	0	0	1	0	1	1	1	4

Table 4.6: Best feature combination for RBF kernel function with C-value$= 0,25$.

In which respect the features considered, one more time eccentricity was selected but this time with the perimeter, Area and I7.

4.2.3.2 Polynomial

Now considering the polynomial Kernel function the results are shown in Figures 4.7 and 4.8.

The best generalization error was obtained for the second order polynomial function, with value 95.6% with the selected features presented in 4.7.

Features Selection:											No. of Features:
I1	I2	I3	I4	I5	I6	I7	I8	Area	Perimeter	Eccentricity	
0	0	0	0	0	0	0	0	1	0	1	2

Table 4.7: Best feature combination for polynomial kernel function with C-value $= 0,25$.

Comparing the performance between the two kernel functions, see graphs 4.6 and 4.8, it is possible to infer that the polynomial function offers a better global result, than those that are provided by the RBF functions. However, for both kernels the soft margin parameter (C-value) is equal to 0.25.

4.3 Conclusions

In this Chapter it was compared different machine learning methods, like k-Nearest Neighbour, Neural Networks and Support Vector Machine, to identify and recognize the objects portrayed in the image. For the work it was used 7 different objects (some of them very similar and more difficult to differentiate) and a feature extraction algorithm based on the invariant moments was applied.

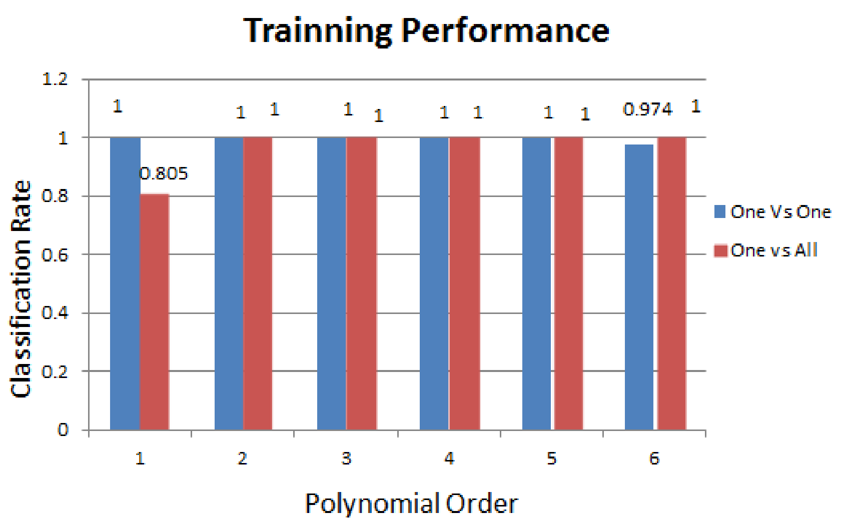


Figure 4.7: Training performance obtained as a function of polynomial degree.

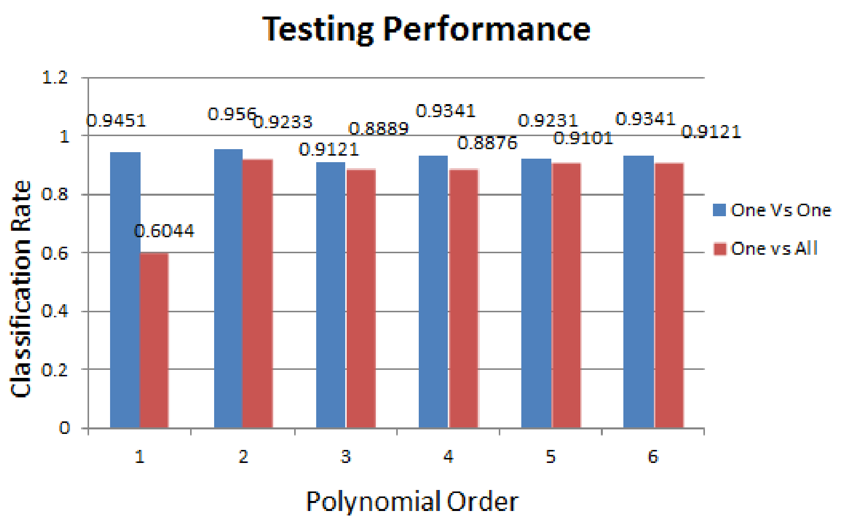


Figure 4.8: Testing performance obtained as a function of polynomial degree.

As it can be verified, it was obtained performances of 83.5% for kNN, 95.5% for NN and 98.9% for SVM (generalized error). These high performances are largely due to the feature selection algorithm based on simulated annealing algorithm and model assessment (K-fold cross-validation) since, it allows the identification of the most important features in the recognition process as well as the adjustment of the best parameter to the classification methods.

The features Area and Eccentricity are present in almost all the results, and based on this fact it is possible to claim that these features are the most important for the classification process (NN and SVM).

It is important to note that, the results obtained were achieved by using MATLAB development environment and respective toolboxes.

All the work presented in Chapter 3 and 4 originated a paper in the International Journal of Robotics and Computer-Integrated Manufacturing [54].

Chapter 5

Final Architecture - Cascade of Algorithms

This Chapter presents the complete architecture for object recognition and pose estimation. This system is based on the 3D modelling and recognition techniques presented in the previous Chapters.

Remember that the challenge for this research work was to create a generic object recognition and pose estimation algorithm to aid industrial manipulators in their task. This algorithm must have the ability to point out to the industrial robot the work-piece Id. and pose, in order to enable a fast and flexible reconfiguration. With this objective in mind, the previous Chapters presented a comparison between some machine learning algorithms. In this comparison, SVMs presented the best results for the classification of 7 different objects. Despite these good results, the classification ratio achieved is not sufficient for an industrial application. These often require close to 100% classification rate, otherwise a misclassification can mean a spoiling part.

Furthermore, previous Chapters also introduced a 3D matching algorithm which can be effectively used for object recognition purposes. The proposed object recognition and pose estimation system uses all the previous techniques in a cascade layout - SVMs and PM.

Nevertheless, the presented system can be extrapolated to other applications where the 3D data is acquired with other hardware.

For different object recognition applications and/or objects, this architecture may require adjustments: to choose a different set of features, and some SVM parameters' tuning (kernel type, misclassification penalization, etc). Also to refer that, although presented as an object recognition algorithm, PM can be used for object 3D pose estimator based on point clouds.

5.1 Perfect Match as a Classifier and Pose Estimator

3D PM is an algorithm based on the off-line computation of the models' distance matrix and the respective x and y gradients matrices. These steps constitute the training phase of the algorithm, and are executed for each object. All the resulting matrices are organized in a database along with the label of the corresponding object.

During the on-line phase and after acquiring the 3D Model of the object to be classified, the PM algorithm will try to fit this unknown object model to one in the database. This geometric matching is performed by using the RPROP algorithm presented Chapters before. At the end of the process, object classification (match with the minimum error) and pose (x ,y and θ) estimation are returned.

Thus, imagine that in an industrial environment the production starts to have many object types and consequently a high number of models is recorded in the PM database. The problem starts to appear when the number of objects in this database is such that the time for all possible object matching and pose estimation is superior to the desired one, creating a bottleneck in the production process. Note that, the PM processing time is linearly proportional to the number of points in the recorded model and to the number of models in the database. Therefore, it is desirable to have a pre-selection of the models recorded, i.e, create a subset of possible objects where the PM algorithm will be applied.

To overcome this difficulty, the first approach was focused in the manually organization of the data set into bins. These are characterized by a specific or a relevant set of distinctive features (dimension, colour, etc). Thus, having extracted those features of the unknown object to classify, the PM algorithm is applied to the subset of objects where those features best fit.

However, some questions emerged. Although the problem of creating a subset was solved, no control in the number of models in each bin was possible to make, since it was dependent on the features that were chosen to characterize each bin. Consequently, it was not possible to control or limit the processing time with this approach.

The problem started to be a harsh one to solve. In this sense, considering the results already presented, SVM was used as pre-selector. SVM classifiers have a higher computational performance time when compared with PM algorithm, allowing the creation of a subset in an efficient way. Note that, with SVM all objects are considered for the initial subset creation.

5.2 SVM Classifier

Taking into account all these considerations, the SVM classifier will first be applied to the object model that is to be classified. Although SVM training is time expensive it is performed off-line.

One of the possible configurations of SVM is the probability SVM. This outputs the probability of an unknown object to fit in the different classes already trained in the training phase. Making it possible to select the *k* best classes to which the unknown model fits better (PM algorithm will only be applied to this subset of classes). At the same time, SVM already retrieves relevant information concerning object classification. This information can be either ignored (the PM classification is accepted) or a fusion of the results retrieved by both classifiers can be performed.

Recall Chapter 2: SVM training is needed; classes' features must be extracted, and an SVM model must be trained; for that purpose, the architecture presented in Figure 2.15 is used.

5.3 Cascade Architecture

The following sections describe the system architecture: a cascade with SVMs and PM algorithm.

5.3.1 Production Phase - On-Line

Starting by the production phase, imagine that an object (belonging to one of the classes trained) was inserted in the production line. The object will go trough the 3D modelling sensor and the extraction of its 3D model is performed.

The next step is the feature extraction procedure, and the run of the SVM classifier (admit that both the PM and SVM are already trained). The result of the SVM is a vector of probabilistic values that indicate the percentage of the unknown model to fit in the classes trained in SVM.

$$P_{Result_{SVM}} = [P_0, P_1, .., P_n], n \in \mathbb{N}^+ \tag{5.1}$$

With *n* equal to the number of classes trained and recorded in the database.

From this probabilistic vector, and as the first followed approach, the *k* best candidate classes returned by SVM (the *k* classes with the highest probability) were selected. Then, the PM is run for those k possible classes. The result with the lowest fitting error will be accepted as the class of the object (see Figure 5.1).

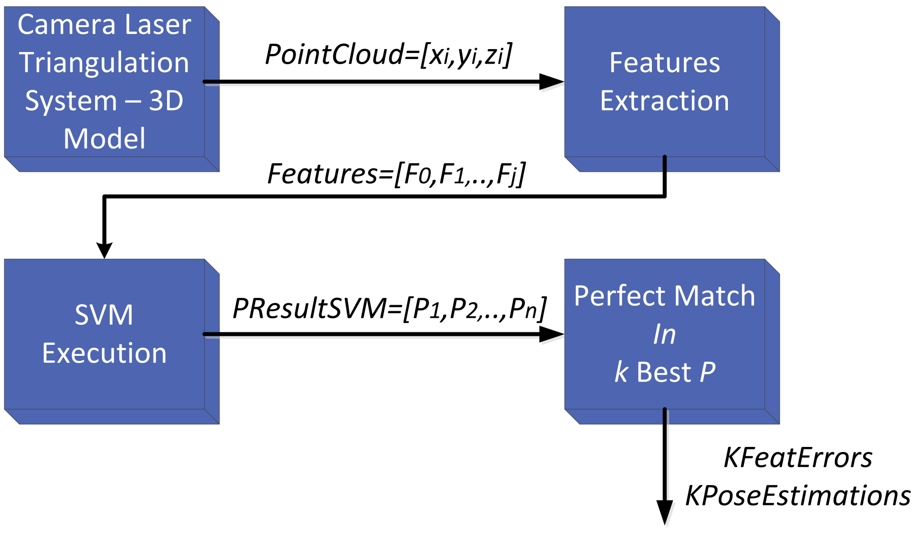


Figure 5.1: Cascade classification architecture.

Although a valid approach, much of the information returned by SVM method is being ignored. Therefore, a condition was created where the SVM result is accepted if that condition is fulfilled, otherwise PM is applied (see Figure 5.2).

In the first case, where SVM classification is accepted, the PM algorithm is only applied for 3 DoF pose estimation - $FeatError$ of the model classified by SVM and $PoseEstimation$ are retrieved. In the case where the SVM classification is not good enough, Perfect Match is applied to k most probable classes, and the classification (best$KFeatError$) and correspondent pose estimation ($KPoseEstimation$) is performed by PM.

The most intuitive condition to verify is to test if the best probability P_k is higher than a specific threshold value. If true, the SVM classification is accepted. This condition will be the first to be tested although more restrictions may be added.

5.3.2 Training Phase - Off-line

Both algorithms are trained independently, although training can be performed in parallel.

Imagine that the user wants to produce a new type of product. To perform the training, this new type of part needs to be introduced in the production chain, and N models are extracted from the new part.

Then, one of the models is used for PM Training (geometric template), that will be used as the model that characterize well all the population. With this model, it is computed

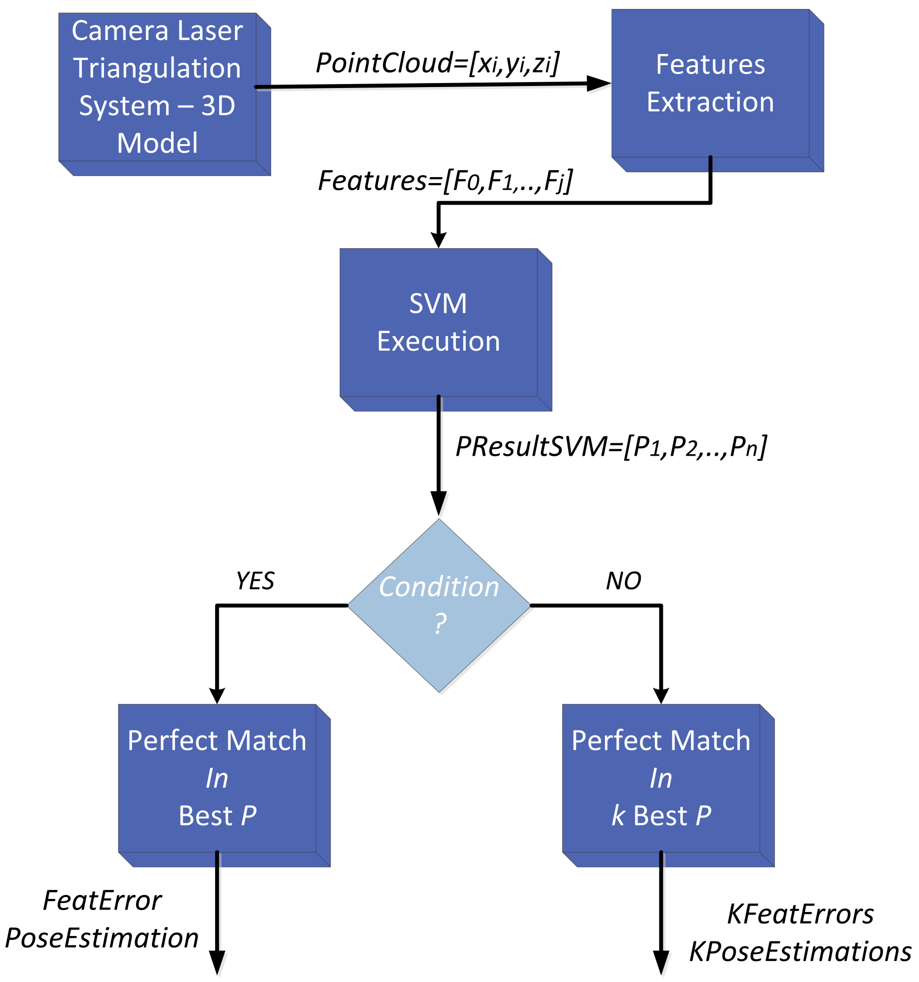


Figure 5.2: Cascade final classification architecture algorithm considered.

the distance matrix and Gradient 3D Matrices already referred before. Note that the robot must know how manipulate this type of object template, since the PM will compute the relative displacement (pose) of the models object to the template.

For SVM training, specific features are extracted from the N models and are saved among the features of other classes. Then all this data is used as the training in the architecture presented in 2.15.

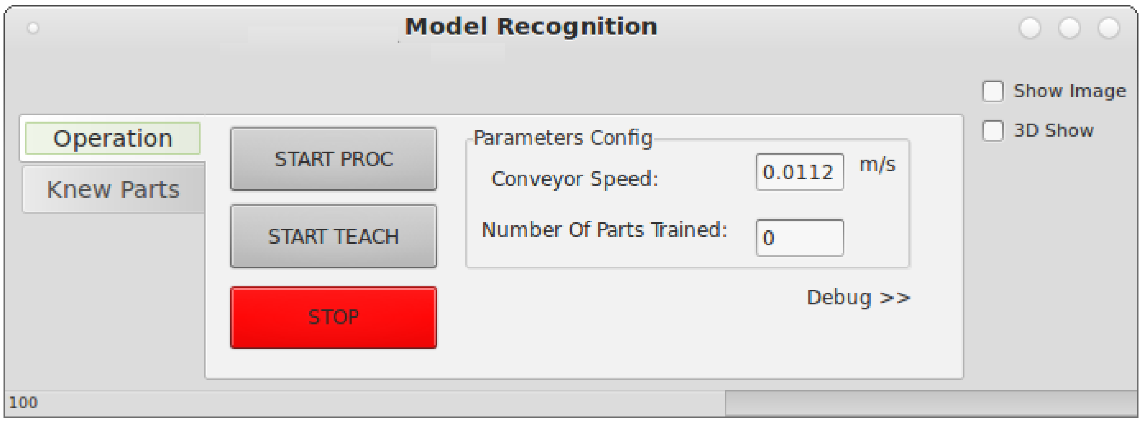
This phase takes some time to perform; however, it is executed in off-line mode.

5.4 Software Developed

To allow this architecture to be tested at an industrial environment, a software application was developed where all the concepts related with object recognition, and discussed until now were integrated.

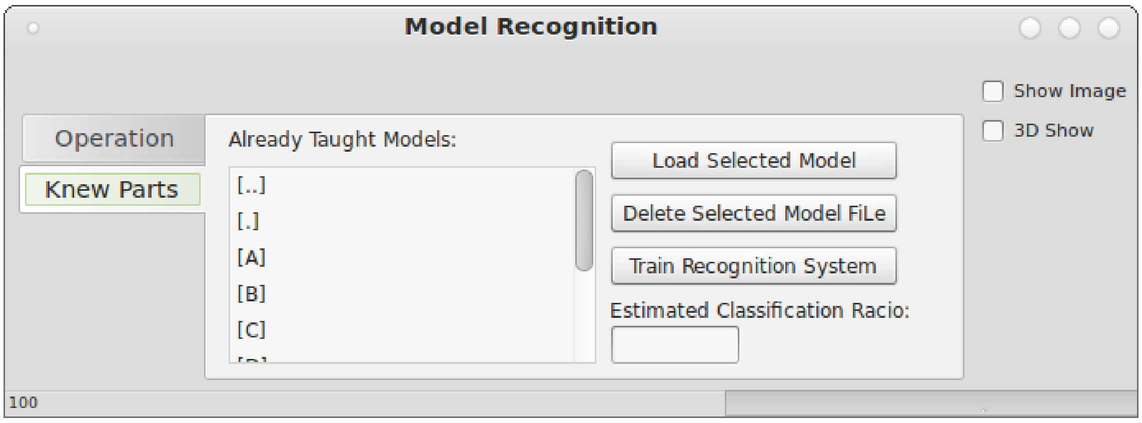
This application has two modes: the *normal* and the *Debug*.

In the *normal* mode, the operator has two tabs: the *Operation* and *KnewParts* (see Figures 5.3 and 5.4).



Parameters Config
Conveyor Speed: 0.0112 m/s
Number Of Parts Trained: 0

Figure 5.3: Software developed - Operators main view.



Already Taught Models:
[..]
[.]
[A]
[B]
[C]

Estimated Classification Racio:

Figure 5.4: Software developed - Parts trained management tab.

In the *Operation* tab, the shop floor operator can select the *production mode* or the *teaching mode*. The former is where the application recognizes the objects ("Start Proc" in Figure 5.3); The later saves the N 3D models so the SVM and the PM algorithm can be trained ("Start Teach" Figure 5.3)).

Considering model training, the first model of each part is considered as the template for PM algorithm, therefore, the robot needs to know how to manipulate this model correctly (in terms of its pose).

In the *Knew Parts* tab, the operator can view the models already trained (Figure 5.5), delete them and rerun all the training algorithm. In the end of the training procedure, an estimated classification ratio is returned.

Finally to consider the *debug* mode. This mode is intended for debugging the application. The most important tab is the off-line where the models for SVM training,

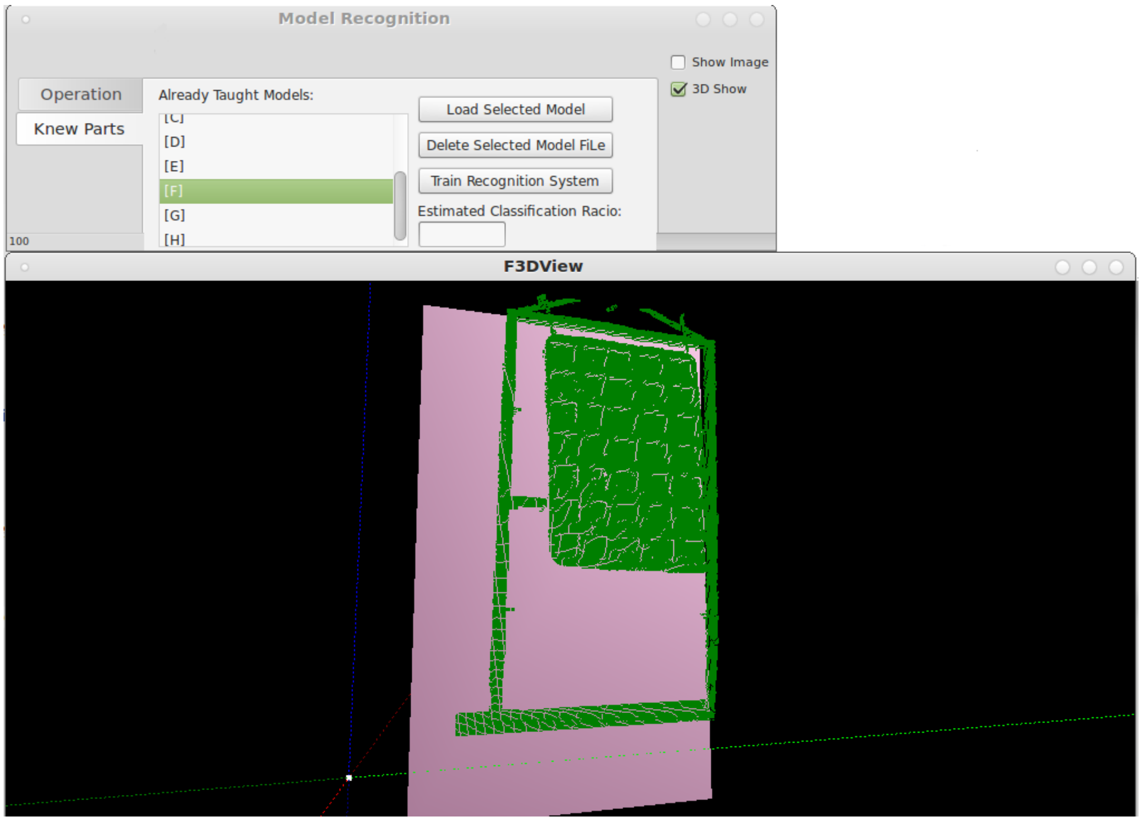


Figure 5.5: Software developed - Operator interface example of a part 3D model.

the PM templates as well as the testing data (used to compute the generalization error) can be analysed independently. In this case, the PM algorithm and the SVM can be run separately, estimating their performances. Therefore, if any problem related with new type of models introduced in the production chain occur, the debug is performed in a simple manner.

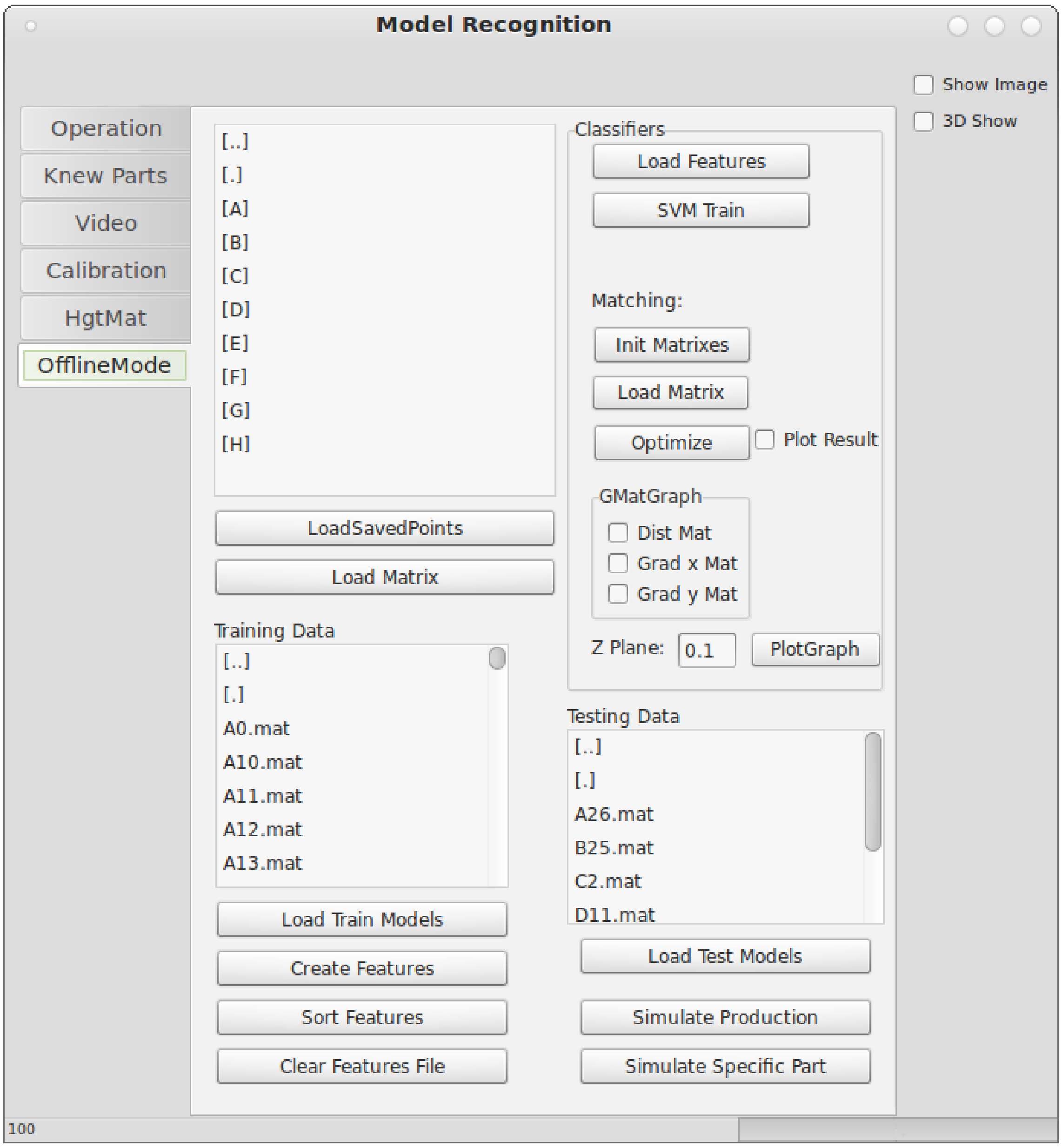


Figure 5.6: Software Developed - Debug mode window.

Chapter 6

Industrial Scenario Results

In this Chapter the constructed object recognition and pose estimation architecture is tested at an industrial demonstrator.

The Chapter is organized as follows: it starts by presenting the industrial partner and its shop-floor environment. Then, it is presented the objects target of 3D modelling. Finally, it is shown the results for the feature extraction procedure, for the object classification as well as for the position estimation.

6.1 Industrial Partner

During the development of the research work, an enterprise has always demonstrated its interest in the solution and the integration on its industrial process. The industrial partner that made part of this work is FLUPOL [1]. FLUPOL is an industrial coating applicator, whose goal is cooperating in solving problems of surface adhesion, dry lubrication or corrosion. Their demands of processes require a very high degree of specialization of their coating operators (more than 10 years training), as well as high flexibility of the means of production given the huge range of different products that are treated.

Today, FLUPOL is focused on the development of a robotized cell that allows a specialized operator to teach (by demonstration) its industrial robot. Beyond this problem, they expect the system to have the capability to identify the object type that has to be coated allowing the robot to upload autonomously the correct program to execute.

The production line in this Portuguese company is characterized by a low speed conveyor line (0.01 m/s) where the products are transported vertically and where the coating operations and heat treatment are applied. Furthermore, each product can go through these two operations several times without leaving the conveyor (the conveyor

[1]FLUPOL - http://www.flupol.pt/

is a closed loop). This production procedure makes it impossible to use sensors like RFID for part identification. Plus, the possibility of the system to have immunity to pose changes (position and rotation) of parts would be a major achievement, due to their poor mechanical conveyor structure.

The approach based on objects' CAD models is conditioned, since neither FLUPOL, nor their clients always have available the parts CAD.

By analysing Figure 3.13, distinguishing object by colour/texture features is out of the possible approaches.

In Figure 6.1 it is presented the overall problem concept. As it is possible to see, the objective is to create an industrial robot complementary system that is responsible for object recognition and pose estimation. This information is then sent to the robot so it can select and perform the coating trajectory correctly. Also note that the industrial robot is only responsible for the execution of its main tasks (coating, manipulation/handling etc).

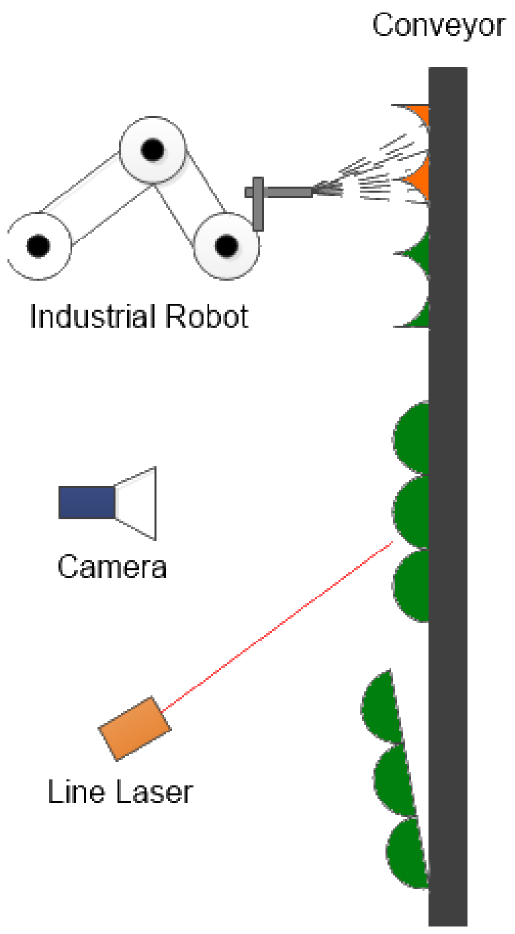


Figure 6.1: FLUPOL problem illustration.

The laboratory prototype presented in Chapter 3 - Figure 3.14 was already built at FLUPOL, allowing to run some preliminary tests, made by Marcos Ferreira et al. [16].

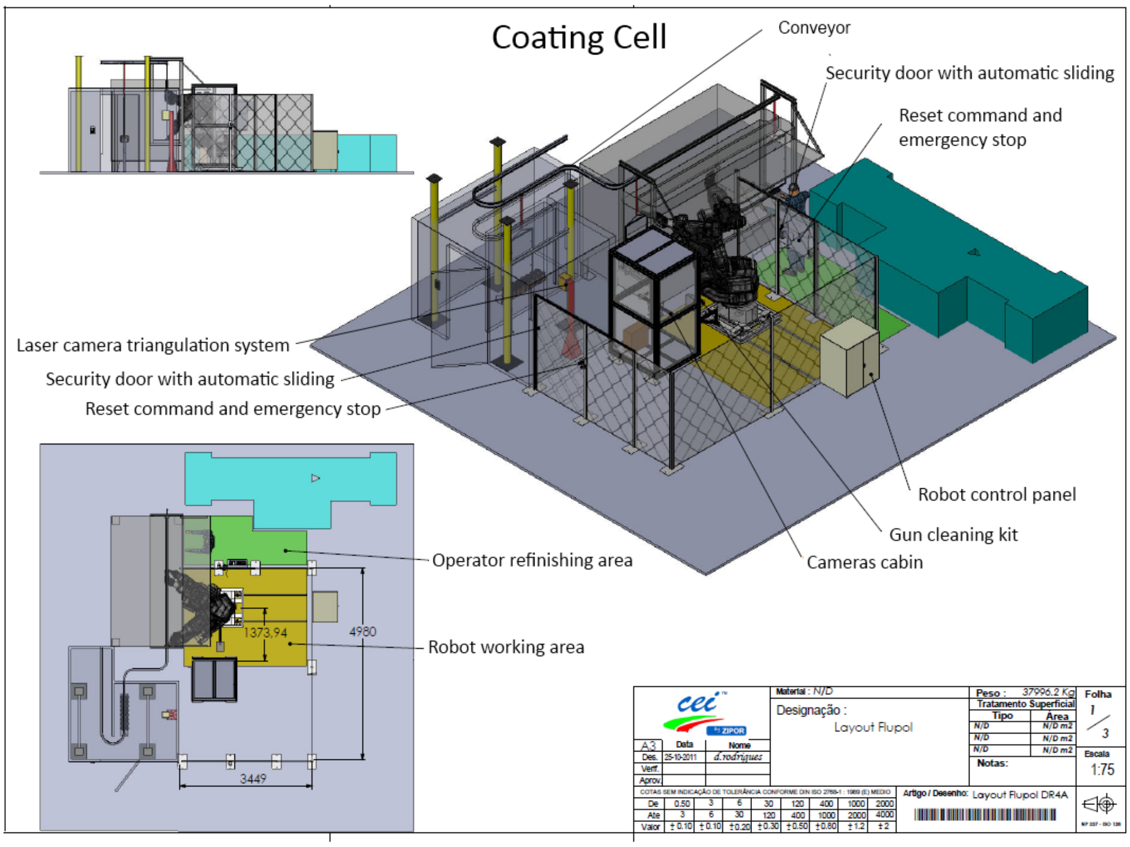

Coating Cell
Conveyor
Security door with automatic sliding
Reset command and emergency stop
Laser camera triangulation system
Security door with automatic sliding
Reset command and emergency stop
Robot control panel
Gun cleaning kit
Cameras cabin
Operator refinishing area
Robot working area
1373,94
4980
3449
Material : N/D
Designação :
Layout Flupol
Peso : 37996,2 Kg
Folha 1/3
Escala 1:75
Des. 25-10-2011 d.rodrigues
Artigo / Desenho: Layout Flupol DR4A

Figure 6.2: FLUPOL shop-floor cell schematic [2].

These preliminary tests in combination with all the work described in this thesis led to the development of a robotic cell prototype that was built at FLUPOL (see Figure 6.2 for the respective schematic). In this prototype, and in what concerns the object recognition application, a dark cabin was built in which the CCD Camera and Laser beam sensors were integrated. Refresh that the 3D modelling solution (based on a Camera-Laser Triangulation) was selected, hence structured light environment needed to be created.

Figure 6.3 shows this cabin and the respective sensors.

The construction of the dark cabin in FLUPOL made it possible to extract new object models at industrial environment.

[2]This Scheme was developed by CEI by Zipor - http://www.zipor.com/.

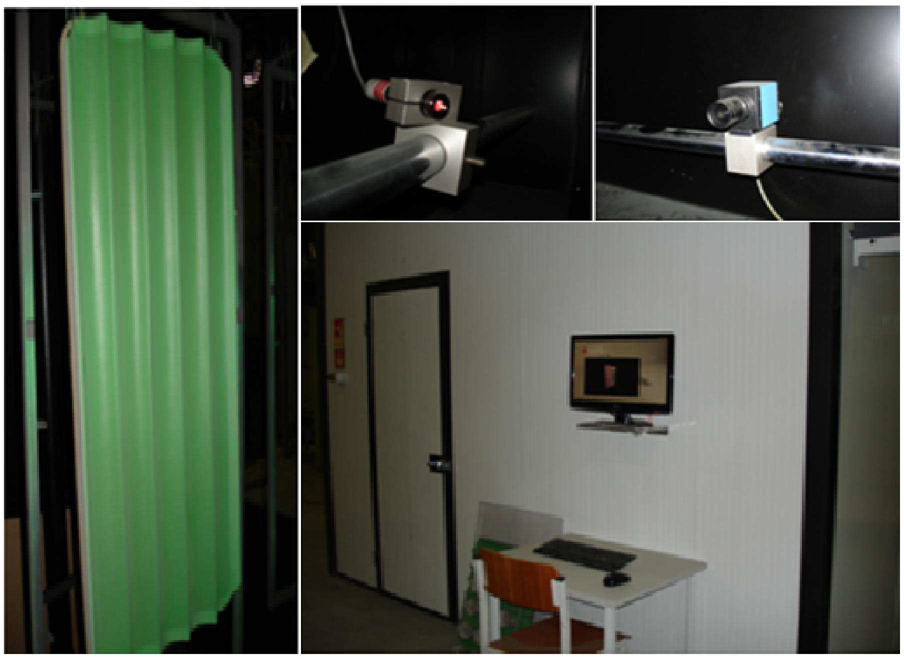


Figure 6.3: Cabin, Sensors, and part example.

6.2 Reconstructed 3D Models - Examples

With a fully calibrated system eight 3D Models of different object classes were extracted and saved in a local database. These models are shown in Figures 6.4, 6.5, 6.6 and 6.7.

Recall that a guide was placed so that the objects' support moves parallel to the Camera-Laser triangulation system. It also helps preventing oscillation of the support in the normal axis to the guider. On the left side of each of the referred Figures it is presented the real object to recognize, and in the right side the 3D modelled retrieved by the Camera-Laser triangulation system.

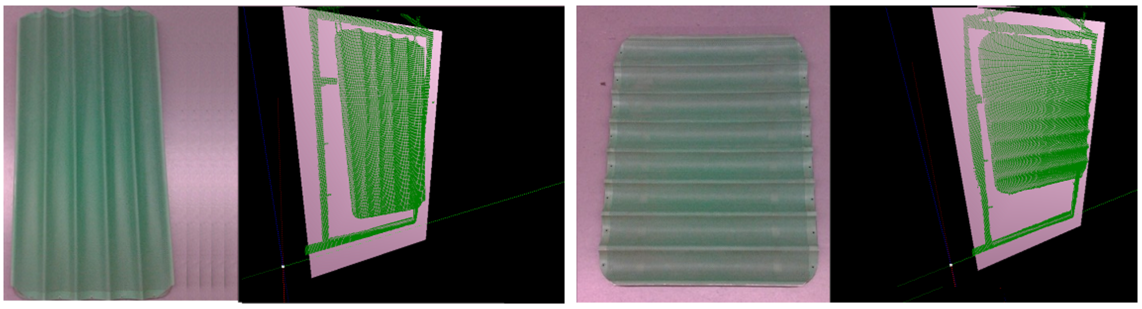


Figure 6.4: Structure of the image created by scanning (Left: class A; Right: Class B).

From the presented Figures, 3D models with low level of noise (robustness to colour, reflection and shape object properties) were obtained, allowing the extraction of 50 samples for each of the 8 classes. These will be split in training and testing data.

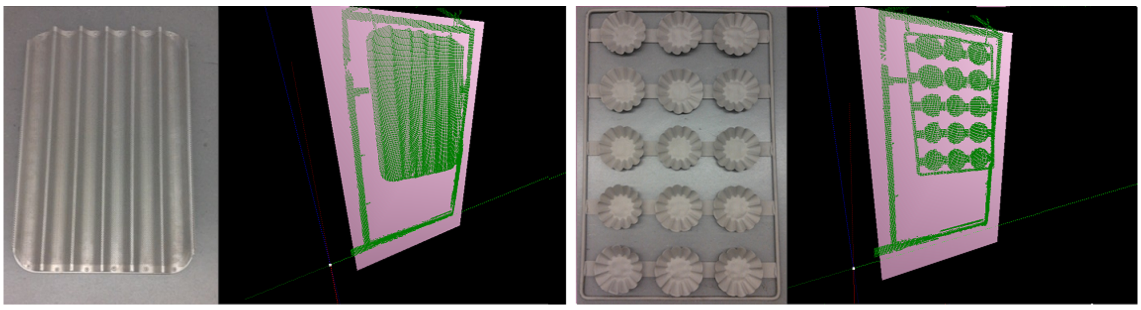


Figure 6.5: Using the laser distances to characterize the environment (Left: class C; Right: Class D).

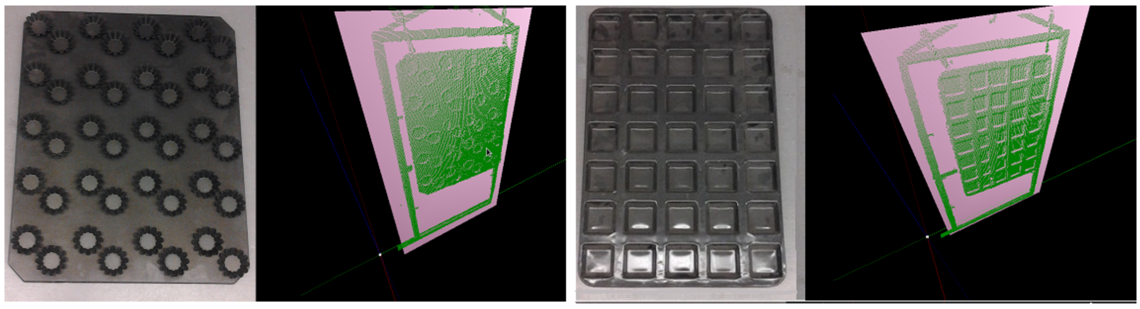


Figure 6.6: Using the laser distances to characterize the environment (Left: class E; Right: Class F).

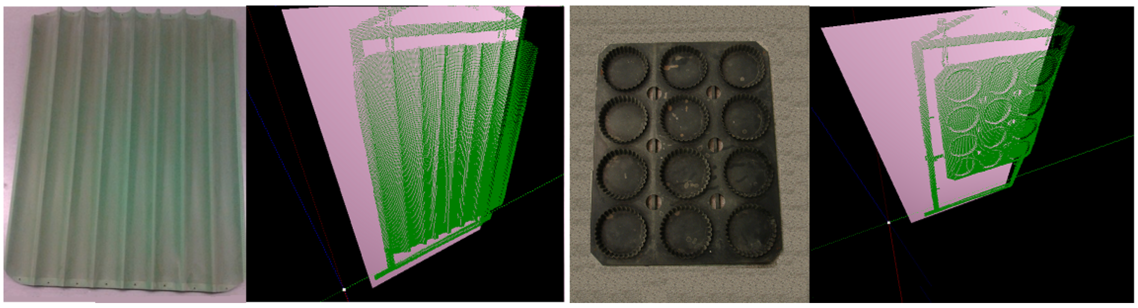


Figure 6.7: Using the laser distances to characterize the environment (Left: class G; Right: Class H).

During the extraction of these models, a problem emerged. The reflective properties of the parts of interest change during the production process. Initially the parts have high-reflectivity (parts are metallic) and after being applied the first coating layer their reflective properties change. This problem has direct influence in the quality of the objects 3D model (imposed by the Laser-Camera Triangulation system). Therefore, this problem was solved at the level of the 3D modelling sensor system, by analysing the width of the laser line. In this sense, and during the production process, the exposure value of the camera is dynamically changed so the width of the laser line is maintained between a given range. Furthermore, the construction of the dark cabin contributed to the good efficiency of the presented solution. This procedure, allowed to have immunity to the variation of the parts reflective properties (imposed by the colour, and specific paint characteristics).

6.3 World Grid Map - Creating the Object Models

For the construction of the occupancy grid map it was considered:

- a maximum height of 1,10 meters
- a maximum with of 0,65 meters
- a maximum depth of 0,40 meters

So, considering these maximum dimensions and a resolution of 2.2 mmm per cell, the 3D model matrix dimensions has $296 \times 500 \times 182$ cells along $width \times height \times depth$ respectively. This size was chosen due to RAM memory limitations and the computational performance of the cascade algorithm. However, care was taken not to deteriorate heavily the 3D data.

In this way all the object models presented in the last section, distance map and gradient map used in the PM algorithm were constructed with these dimensions.

6.4 Parts Classification

Having extracted the samples, the next step is to train the recognition algorithms. For the training of SVM, features have to be extracted and the SVM model needs to be trained. Therefore, the features and also the training architecture presented in Chapter 2 are used. On what concerns the PM algorithm, a model that represents well the entire population (geometric template) needs to be recorded. Also, the distance and gradient matrices must be computed.

6.4.1 SVM Training

Starting with the SVM, the first step aims to simplify the differentiation between each model. As such, the object support (carrier) was removed from the 3D model because it is present in every sample, hence not helping in distinguishing them.

6.4.1.1 Removing the Support

To remove the support a sample was recorded of this part alone (with no part being held). Then, the best plane that approximates better all the points belonging to this empty support was computed (Figure 6.8).

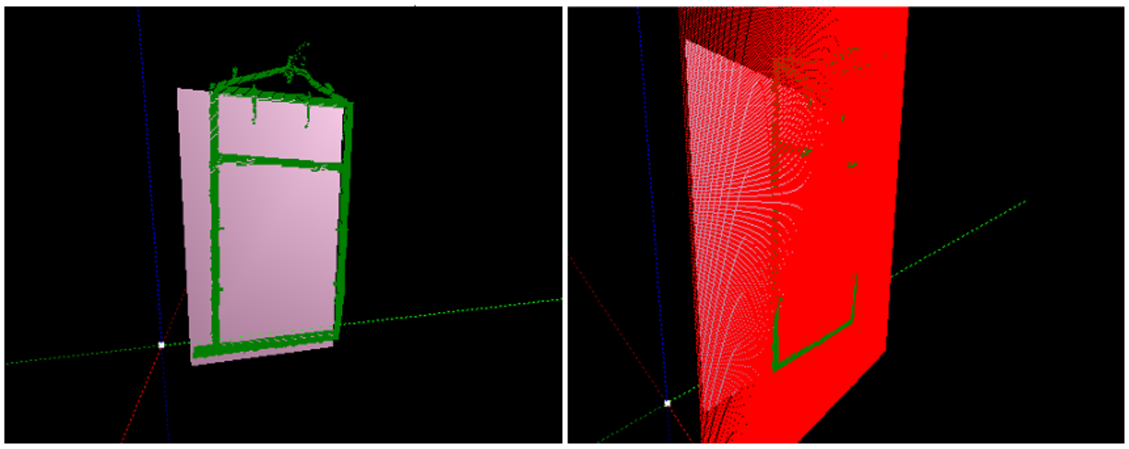


Figure 6.8: In this Figure it is presented the support and the correspondent plane approximation.

Both during the SVM training and the production phase the support was removed from every object model. Figure 6.9 shows an object example where the support was removed with success.

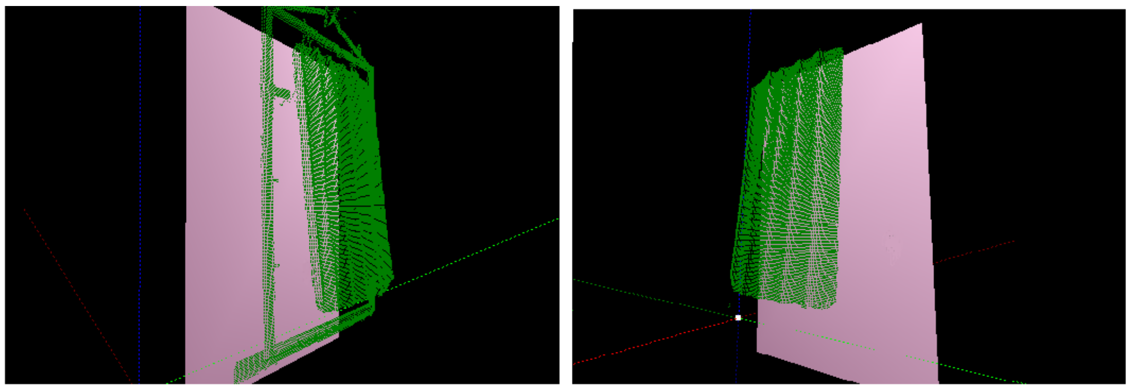


Figure 6.9: Example of a part support removal.

This removal was performed by deleting the points closer to the plain computed before.

6.4.1.2 Computing the object boundaries

The next step was directed to the computation of the object boundaries and 2D horizontal and vertical cuts, that will be useful for feature extraction.

The extraction of object boundaries (after the support removal) is trivial to get. The following search in the world grid map is performed:

- For every line of the 3D Matrix with depth different from zero, the minimum and maximum indices (column) are saved. This defines the left and right boundaries of the object.
- For every column of the 3D Matrix with depth different from zero, the minimum and maximum indices (lines) were saved. This defines the upper and bottom limits.

The result is presented in Figure 6.10.

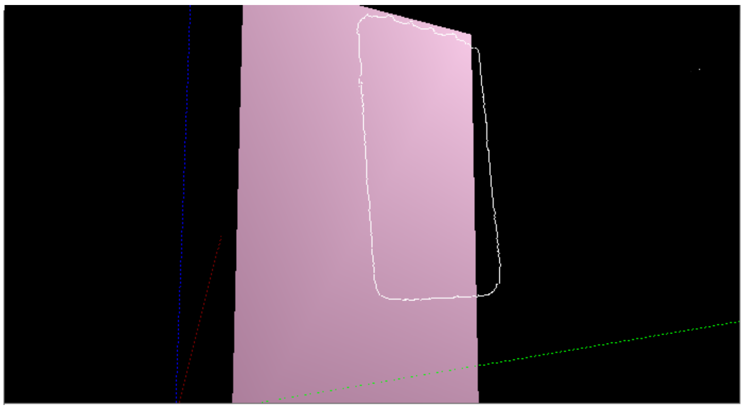


Figure 6.10: Example of boundaries computation.

After the computation of object boundaries, the dimensions are easy to compute - the distance between the limits of the part (the mean for all rows and columns). The height and width are the first two features used by the classifier.

Another important data that is extracted from the 3D models are the horizontal and vertical cuts. FFT will later (next section) be applied to these cuts so an analysis of the pattern in the referred directions can be performed. For each model, and to consider the respective boundaries, 15 cuts were taken for each direction, as shown in Figure 6.11. FFT magnitude and spatial frequency index will be saved performing in total 60 features.

6.4.1.3 Fast Fourier Transformation

As referred, Fast Fourier Transform was used to analyse the pattern along the objects' width and height. Figure 6.9 presents the FFT graphs for a horizontal, and vertical cut of the object considered until now for feature analysis.

In this way it is expected that the FFT for each horizontal cut to have an amplitude peak different from zero in a spacial frequency. As for the vertical cuts, it is expectable that all amplitude peak values be near frequency zero.

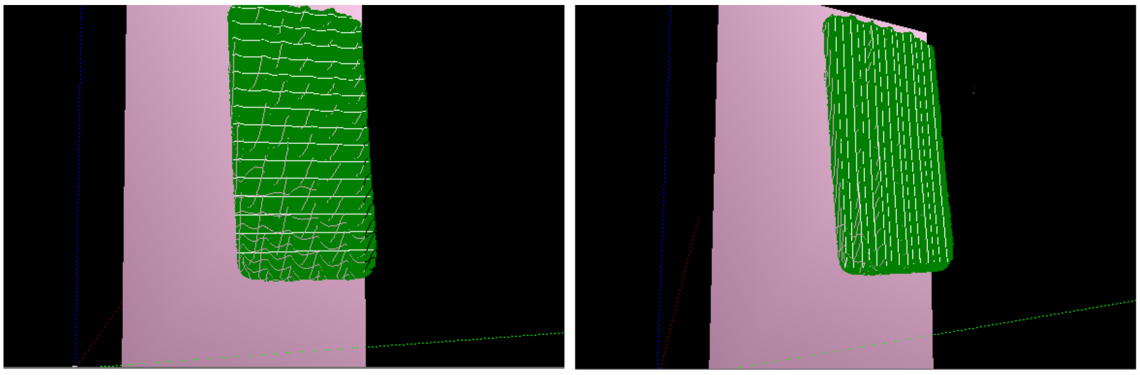


Figure 6.11: Objects Horizontal and Vertical Cut.

Starting by the application of the FFT directly to a vertical and horizontal cuts, figures 6.12 and 6.13 present the respective results.

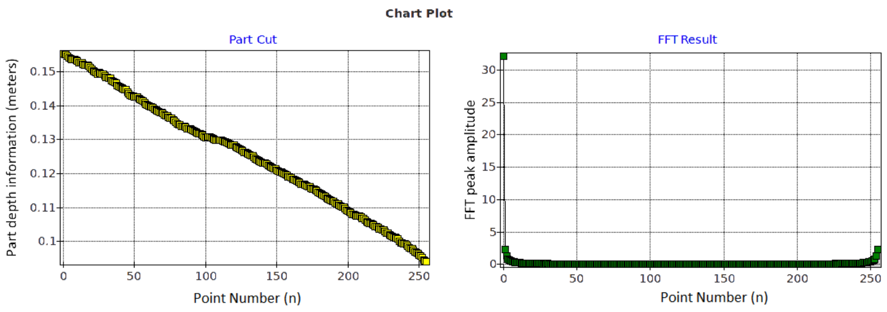


Figure 6.12: FFT applied to a vertical cut.

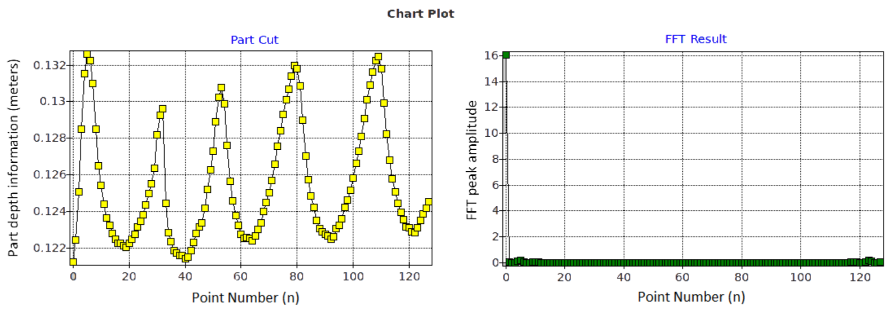


Figure 6.13: FFT applied to a horizontal cut.

As it is possible to see, the cuts are strongly affected by the DC component (well illustrated in the FFT results). In Figure 6.13, apart from the low frequency peaks, it is also possible to detect small amplitude peaks that illustrate the frequency of the horizontal cut.

Therefore, starting by removing the DC component seems to be a good strategy. However, note that both of the cuts are not centred in the horizontal axis. This is due to the fact that the part is not completely vertical and parallel to the Camera-Laser triangulation system. Therefore for removing the DC component it is necessary to enter with this variable. The solution passes to approximate the cut by a linear regression and compute the bias from there.

Figures 6.14 and 6.15 presents the cuts with the DC component removed and the respective FFT result.

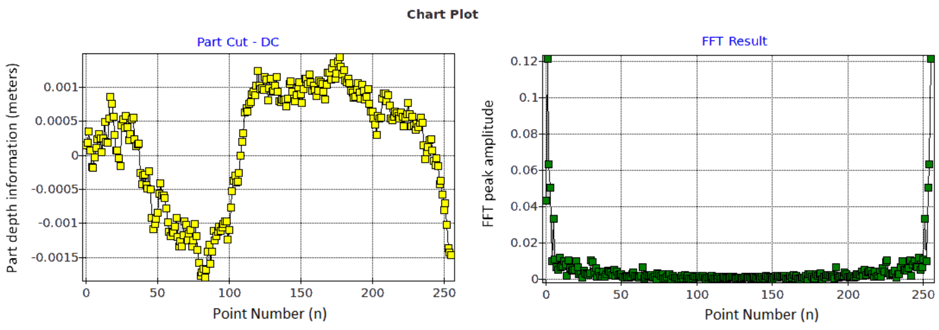


Figure 6.14: FFT applied to a vertical with DC component removed.

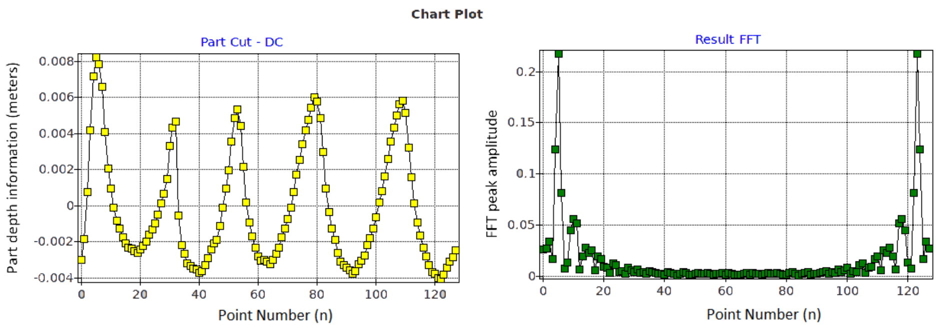


Figure 6.15: FFT applied to a horizontal with DC component removed.

By looking only to the FFT results and comparing with the previous obtained (with DC), it is possible to see that the amplitude value of the low frequencies have suffered a high peak value reduction as expected. Moreover, in Figure 6.15, it is clearly identified the amplitude peak resultant from the horizontal cut pattern.

Finally, and dealing with the leakage problem presented chapters before, the cut has been multiplied by a window function (Hamming Window). This aims to low the spread of the fundamental amplitude over a wide range of frequencies.

By analysing Figures 6.16 and 6.17 it is possible to conclude that the energy of the signal (cuts) is more concentrated in a specific range as it was desired. This enables a clear identification of the cuts' fundamental frequency.

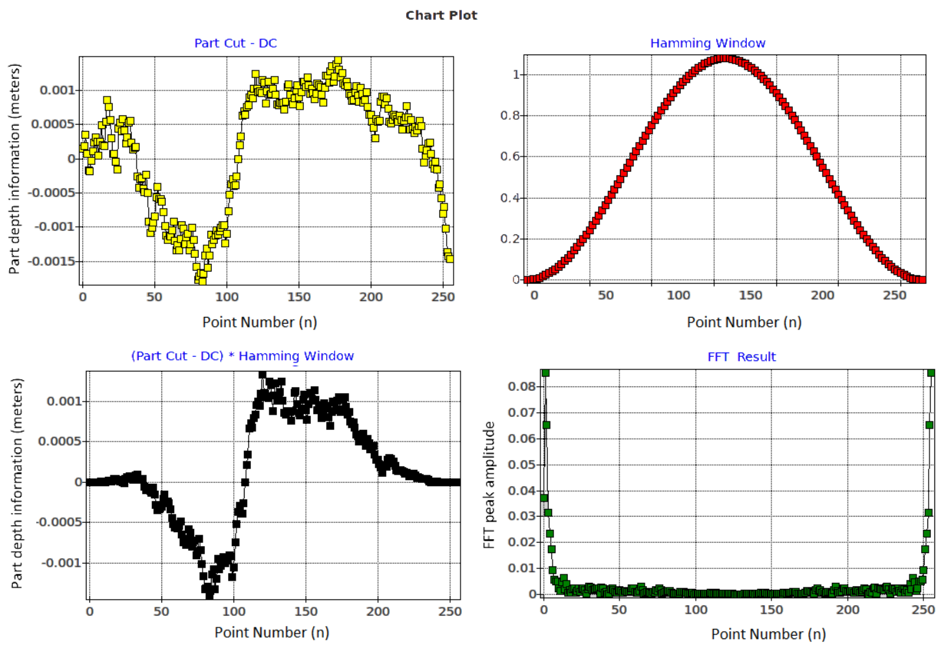


Figure 6.16: FFT applied to a vertical with DC component removed and leakage compensation.

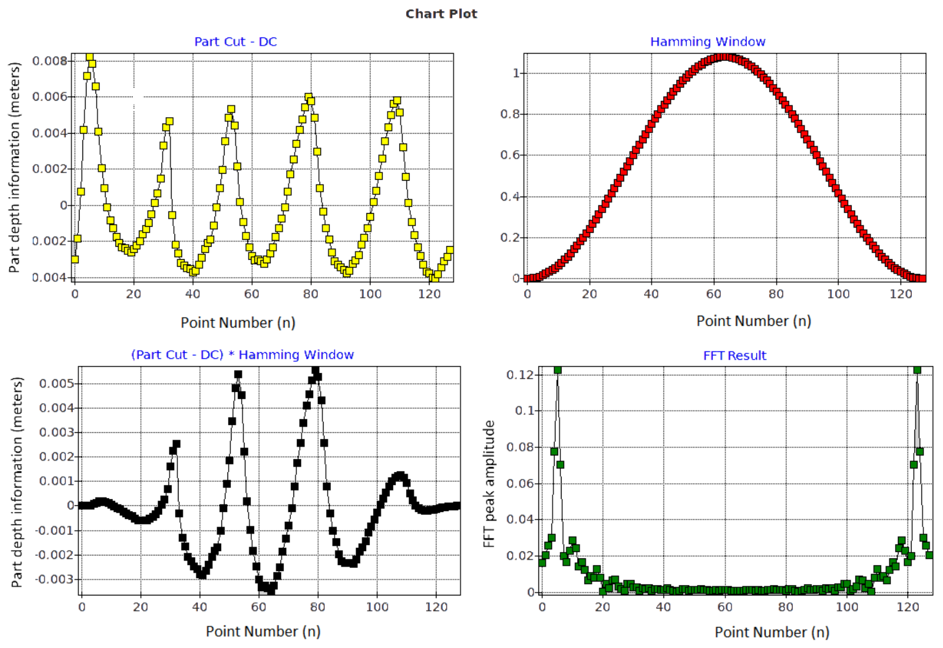


Figure 6.17: FFT applied to a horizontal with DC component removed and leakage compensation.

Just to clarify, to get extract the geometric frequency (Hz) of the FFT, the following expression is applied:

$$n \times \frac{Fs}{N}. \tag{6.1}$$

where n is the point/bin number, Fs the sampling frequency (that in this case is $1/(3D\_VoxelResolution)$), and N the total number of points/bins.

6.4.1.4 8 Hu Moments

For the computation of the Hu Moments the models were transformed into a grey-scale image, so the features could be computed. For that, it was considered an object's maximum depth of 400 mm (255 in a single byte grey-scale) and a minimum of 0 (0 in grey-scale).

From this grey-scale image it is computed the 8 Hu Moments, area, eccentricity, and the energy and entropy of the image's histogram (equations and definitions can be found at Chapter 2).

6.4.1.5 Feature Summary

In summary it is computed the following features:

- 60 features from FFT (15 horizontal and 15 vertical cuts characterized by the magnitude and frequency of the maximum peak);
- The 8 Hu Moments;
- Object Width and Height;
- Area and Eccentricity;
- Histogram Energy and Entropy.

There are a total of 74 features.

6.4.2 SVM Model Computation and Generalization Error Results

Now that features were extracted from the different 3D object models, the creation of the SVM model for classification purpose is necessary. Therefore, beyond extracting various 3D models for training SVM classifier, parameters tuning needs to be assessed.

LibSVM was the selected library used for implementing the SVM Classifier [3]. It deals with multi-class problems using a one-against-all approach.

For classification purposes, the SVM parameters needing tuning are:

- kernel type and respective parameters.

[3]http://www.csie.ntu.edu.tw/ cjlin/libsvm/

- Cost value (C) - parameter that controls the trade-off between the double objective of maximizing the margin of separation and minimizing the misclassification error.

There are several kernel types: linear, polynomial, radial basis function and sigmoid. Remember that the kernels were presented in Chapter 2. Considering these good results obtained with SVM polynomial or radial basis kernel, these two approaches were the selected ones for the present classification problem. Each type needs different tuning:

- for polynomial kernels it is necessary to tune the degree and gamma (γ) parameters.
- for the radial basis it is necessary to tune only the gamma (γ) parameter.

All classification tests will be conducted with the architecture presented in Figure 2.15. Thus, the data was divided into training $Data1_{n\times m}$ (n is equal to 35 samples of each class and m equal to 74 (number of features)) and testing data $Data2_{i\times m}$ (i is equal to 15 samples of each class and m equal to 74 (number of features)).

For the first iteration of the simulated annealing step, only 15 out of the 74 features were considered as active (used for the SVM model training). This subset of features was selected considering initial results. The objective with this is to try to find the best minimum number subset of features (decrease model complexity), for which the SVM model achieves maximum training accuracy.

6.4.2.1 Results for Radial Basis Kernel Function

Considering the Radial basis function kernel defined as:

$$k(x_1,x_2) = exp(-\gamma\|x_1 - x_2\|^2) \tag{6.2}$$

The only parameter necessary to tune is the parameter γ (gamma). Therefore, for the results presented next, the γ parameter was varied considering the following equation:

$$\gamma = \frac{i}{NumberOfFeatures} \tag{6.3}$$

Where i can assume the following values: 1,10,100 and 1000.

Furthermore, the C value (soft margin parameter) will also need to be tuned. The range of this variable was defined as [4,7] after some initial tests.

To tune this parameter K-fold cross-validation cycle was used. Therefore, in each iteration of simulated annealing, where the features considered are varied ($NumberOfFeatures$ not constant), an SVM model is trained with a combination of

γ parameter, and C values possible. The training accuracy of these models is recorded, and the best one is chosen.

In Tables 6.1, 6.2 and 6.3 the results considering the discussed kernel are presented.

Simulated Annealing Iteration	Accuracy Value
1	0.935
2	0.940
3	0.960
17	0.970
44	0.975
51	0.980

Table 6.1: Training accuracy results with the increase of the simulating annealing iterations, for radial basis kernel function.

Simulated Annealing Iteration	C Parameter Value
1	6.75
2	7.00
3	7.00
17	6.75
44	5.25
51	5.25

Table 6.2: Training - C value results with the increase of simulating annealing iterations.

Simulated Annealing Iteration	i Parameter Value
5	1
10	100
14	1
21	10
55	1

Table 6.3: Training - i ($\gamma = i/NumberOfFeatures$) value results with the increase of simulating annealing iterations.

Table 6.1 presents the evolution of the estimated training accuracy value with the increase of iterations of Simulated Annealing. Note that table 6.1 shows an increase of accuracy which validates its functionality; yet, a poor solution may have been accepted so that a better accuracy value is achieved at the end (which is intrinsic to the simulated annealing process).

Tables 6.2 and 6.3 present the best values returned by K-fold cross-validation for the C and i parameter for each iteration of the simulated annealing. This best classification was achieved for C (soft margin parameter) value equal to 5.25 (value iterated between 4 and 7), with i ($Gamma = i/NumberOfFeatures$) equal to 1 and with a training accuracy value of 98% (if all the features were considered to build the classifier model, the training accuracy was equal to 96.4 %).

This estimated model was tested in an independent test set with 120 samples of 8 classes, and the generalization error obtained was 92%.

6.4.2.2 Results For Polynomial Kernel Function

This section presents the polynomial kernel which is defined as:

$$k(x1,x2) = (\gamma x_1^T x_2)^{degree} \tag{6.4}$$

For this approach it is necessary to tune the parameter γ, the polynomial degree and the C (soft margin parameter) value.

C varies in the interval [4,6]. γ varies the same way as presented in the previous section for the radial basis function.

Tables 6.4, 6.5 and 6.6 present the results for the variation of the best training accuracy estimated and the corresponding best C and γ values.

Polynomial Order	Accuracy Value
2	0.975
3	0.975
4	0.978
5	0.986

Table 6.4: Training accuracy results for polynomial kernel function.

Polynomial Order	C Parameter Value
2	4.50
3	5.00
4	4.50
5	4.75

Table 6.5: Best C estimated value for each polynomial kernel degree.

Analysing the results, it is possible to conclude that the best training accuracy value (98.6%) was achieved by using C value equal to 4.75, i equal to 10 and polynomial degree equal to 5.

Just to illustrate one more time the performance of the simulated annealing (to select the important features) and the K-fold cross-validation (to select the C and γ parameter).

Polynomial Order	i Parameter Value
2	100
3	1000
4	10
5	10

Table 6.6: Best i estimated value ($\gamma = i/NumberOfFeatures$) for each polynomial kernel degree.

Tables 6.7, 6.8 and 6.9 present the simulated annealing iterations considering a 5 degree polynomial kernel.

Simulated Annealing Iteration	Accuracy Value
0	0.962
6	0.972
10	0.975
16	0.978
17	0.982
54	0.986

Table 6.7: Training accuracy increasing results for polynomial kernel function.

Simulated Annealing Iteration	C Parameter Value
0	6.00
6	5.00
10	6.00
16	4.00
17	5.75
54	4.75

Table 6.8: Best C estimated value with the increase of simulated annealing iterations.

Simulated Annealing Iteration	i Parameter Value
0	100
6	1000
10	1000
16	100
17	100
54	10

Table 6.9: Best i estimated value ($\gamma = i/NumberOfFeatures$) with the increase of simulated annealing iterations.

As it is shown with the increase of simulated annealing iterations a better model is estimated, resulting in a higher training accuracy. At the same time it is possible to see the best C and γ value tuned by K-fold cross-validation for each estimated model in each Simulated Annealing iteration.

If all the features were considered the best accuracy value achieved in the training phase was for polynomial degree equal to 5, with 96.1% of the object correctly classified. This shows the importance of the inclusion of simulated annealing in the architecture for the classifier training.

Finally, in Table 6.10 is shown the generalization error for the different polynomial degrees(using the already tuned SVM models).

Polynomial Order	Accuracy Value
2	0.966
3	0.966
4	0.983
5	0.975

Table 6.10: Generalization error for the different SVM model tuned before.

Analysing the results, it is possible to conclude that the generalization errors are very close to each other, as expected from the results achieved in the training process. The explanation for the polynomial degree equal to 5 giving poorer results than the polynomial degree of 4 is that a data over-fit may have occurred despite the affords by using cross-validation.

Therefore, the best overall result achieved was for the polynomial degree equal to 4 with a generalization accuracy value of 98.3%.

6.4.3 Training the Perfect Match Algorithm

For the PM a local database was created where the information of each part was saved. This information consists of:

- 3D model of the object
- Distance Matrix
- Gradient Matrices along x and y
- Label introduced by the operator

Throughout this Chapter, the 3D model of a specific part has been presented (given as an example). Therefore, the first step is completed.

Next is the computation of the distance matrix in the occupancy grid map.

Therefore, in Figure 6.18 a vertical cut performed in the 3D Distance Matrix is shown. The darker zones correspond to points closer to the 3D model. The brighter ones are the farthest points. As it is possible to see, the cut was performed close to the parts support.

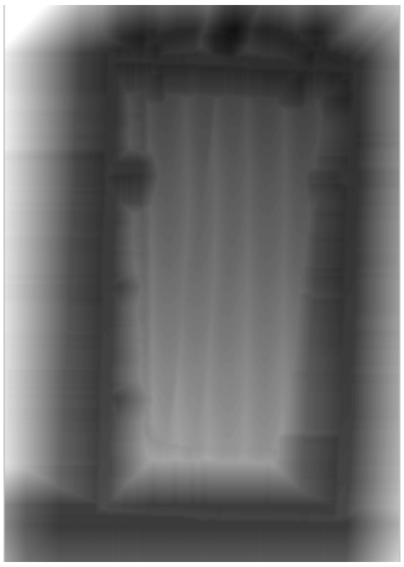


Figure 6.18: Vertical cut of distance matrix.

This image allows the validation of the distance matrix computation for each class model.

Considering the distance map presented before, the correspondent gradient along x and y (vertical cut at the same position) is presented in figure 6.19.

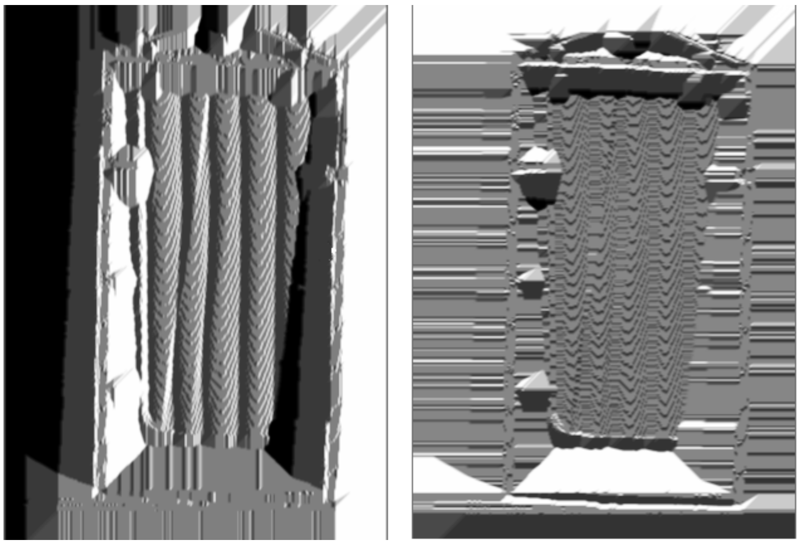


Figure 6.19: Gradient Matrix along x and y.

These three matrices, distance map and gradient matrices along with the object 3D model and respective label are stored in a local database. This procedure is made for all the types of objects that are to be processed in the production line, and are called the templates.

As the last parameter to tune in the PM algorithm is the parameter L_C in the error cost function (equation 2.56). This parameter balances the weight of the distances values to the error function. In other words, it limit the contribution of outliers to the computation of the matching error. Considering the parts size, and possible object translation and rotation, it was defined a L_C value of 0.4 meters.

6.4.3.1 Perfect Match Results

This section describes the classification process of an unknown object (must belong to one of the modelled classes, currently available in the database). Basically, the red part in each of the Figures 6.20 and 6.21 is the unknown 3D model that will be compared with the labelled models saved. The match with the minimum error is the label of this unknown part.

Therefore in Figures 6.20 and 6.21 are presented some visual examples of the 3D matching. Table 6.11 presents the result of applying the matching of an unknown model with all the database. Moreover, the computed values for pose correction are also shown.

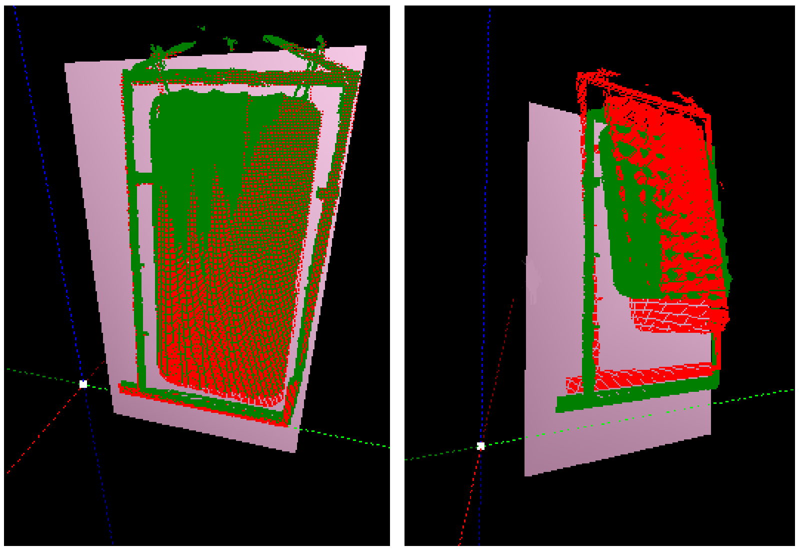


Figure 6.20: Perfect Match Applied to unknown part vs class A and B.

As it possible to conclude from table 6.11, and by analysing the matching error, the best match to the unknown model is class A. Figure 6.22 presents the unknown and the class *A* model recorded in the database.

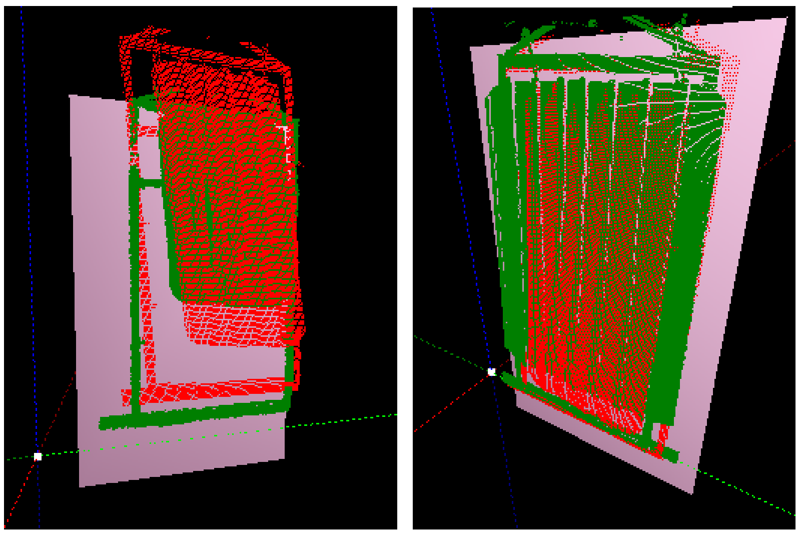


Figure 6.21: Perfect Match Applied to unknown part vs class C and G.

Matching Unknown	Error (E in meters)	x pose correction (meters)	y pose correction (meters)	theta(degrees)
vs. A	3.8143	-0.0043	-0.0129	-0.006
vs. B	30.6859	0.0209	0.0328	4.137
vs. C	54.5575	0.0452	0.0462	5.506
vs. D	39.9767	0.0348	0.0327	4.796
vs. E	68.8505	0.0340	0.0480	3.673
vs. F	61.3389	0.0412	0.0463	3.535
vs. G	23.7696	-0.0301	-0.0129	0.000
vs. H	74.3672	0.0390	0.0391	4.086

Table 6.11: Perfect Match Classification plus pose adjustments.

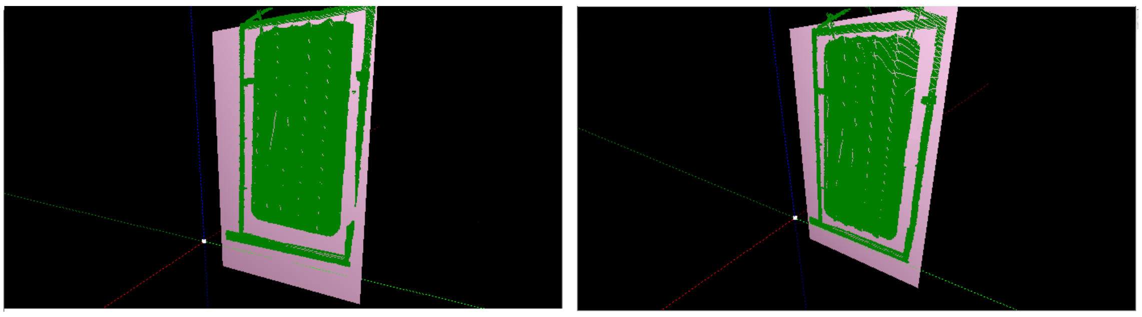


Figure 6.22: Left image - the unknown part model. Right image - the class A recorded model.

In the article published in the conference FAIM 2013 [60], this algorithm was tested for eleven different FLUPOL objects classes. Figure 6.23 presents some of these parts. Other examples were already shown in Figure 6.2.

One sample of each class was used as a template and recorded onto the database. 320 objects were classified using the PM algorithm, achieving a classification rate of 99.7%. The presence of a high amount of noise in the models (in part explained by the

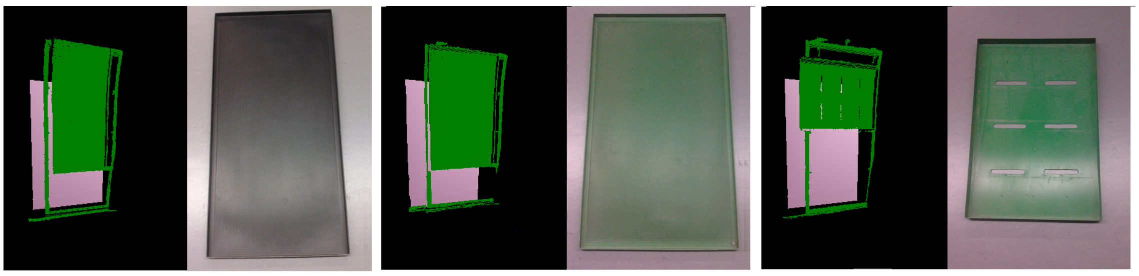


Figure 6.23: Three additional parts considered for the laboratory setup.

CCD Camera 2.4 mm focal length, which presents high level of distortion) increased the difficulty of the classification. This data was acquired using the FLUPOL prototype already presented in Figure 3.14.

6.4.3.2 Processing Time

Although good results have been achieved, one of the major problems is related to the processing time. In this section, a short study is performed on the number of model points and iterations. The results presented in the last section (and also presented in FAIM 2013) were achieved with 100 iterations for RPROP, and using all the points from the data model structure (matrix) to perform the matching. For each matching test the estimated processing time is around 2s (0.5 s to load the distance matrix and gradient matrices, and 1.2s to perform the matching). The loading is related to the size of the matrix and to the precision required for the application. A 296 x 500 x 182 matrix was considered with a 2.2 mm resolution. As previously mentioned, the number of iterations of the RPROP is the parameter that controls the computational speed of the PM. Although one of the important aspects is the processing time, estimating the displacement precision is also a significant task. Therefore, for the FAIM results, the error was minimized with 300 iterations of RPROP with a computational cost of 4s. This is not satisfactory for the actual industrial application. In this way, a down-sampling of 2 in the 3D model was made, achieving matching times of 530 ms. Adding this matching time with the matrices load time the algorithm computational cost is about 1s (tests performed using an Intel Core i7 2.93 GHz).

6.4.4 Results for the Cascade approach

To apply the Cascade approach it is necessary to define the size of the subset returned from probability SVM, which will be applied to PM. This subset was defined with a size equal to 2. Furthermore, the fusion of the SVM results and the PM algorithm was considered.

Considering the subset of objects returned by SVM and the architecture presented in the last Chapter, if the best class returned by the SVM have a probability higher than or equal to 70% and the second best fit class have a probability that is lower than or equal to 6%, the PM is only applied for pose estimation and the label of the unknown object is given by SVM. If the condition is not respected, the two best probabilistic values are selected from the SVM classification tests that correspond to the two best fit class labels of the unknown object. Then, the PM is applied between the unknown model and the two best fit templates estimated by SVM. The result label class is the matching with the lowest error.

With this final architecture and using the same data used to compute the generalisation error in the SVM training phase (8 classes and 120 samples), a 99.2% accuracy was achieved. The error in classification was due to a higher rotation of the object in the width axis (y), that is not estimated by PM. Also, for the SVM procedure and due to the support removal step, the object was "deformed" (a part of the object of interest was cut) having direct impact in the features computation. Furthermore, PM could not get this intrinsic rotation and performed a best fit with other similar object. However, this defect in the production line will be corrected by imposing structural/mechanical constraints. This will prevent the parts rotation, allowing the achievement of 100 % in classification.

As it is possible to conclude, the classification rate with the cascade architecture increased when comparing the results using only the SVM classifier. Figures 6.24 and 6.25 present interesting results obtained by this approach.

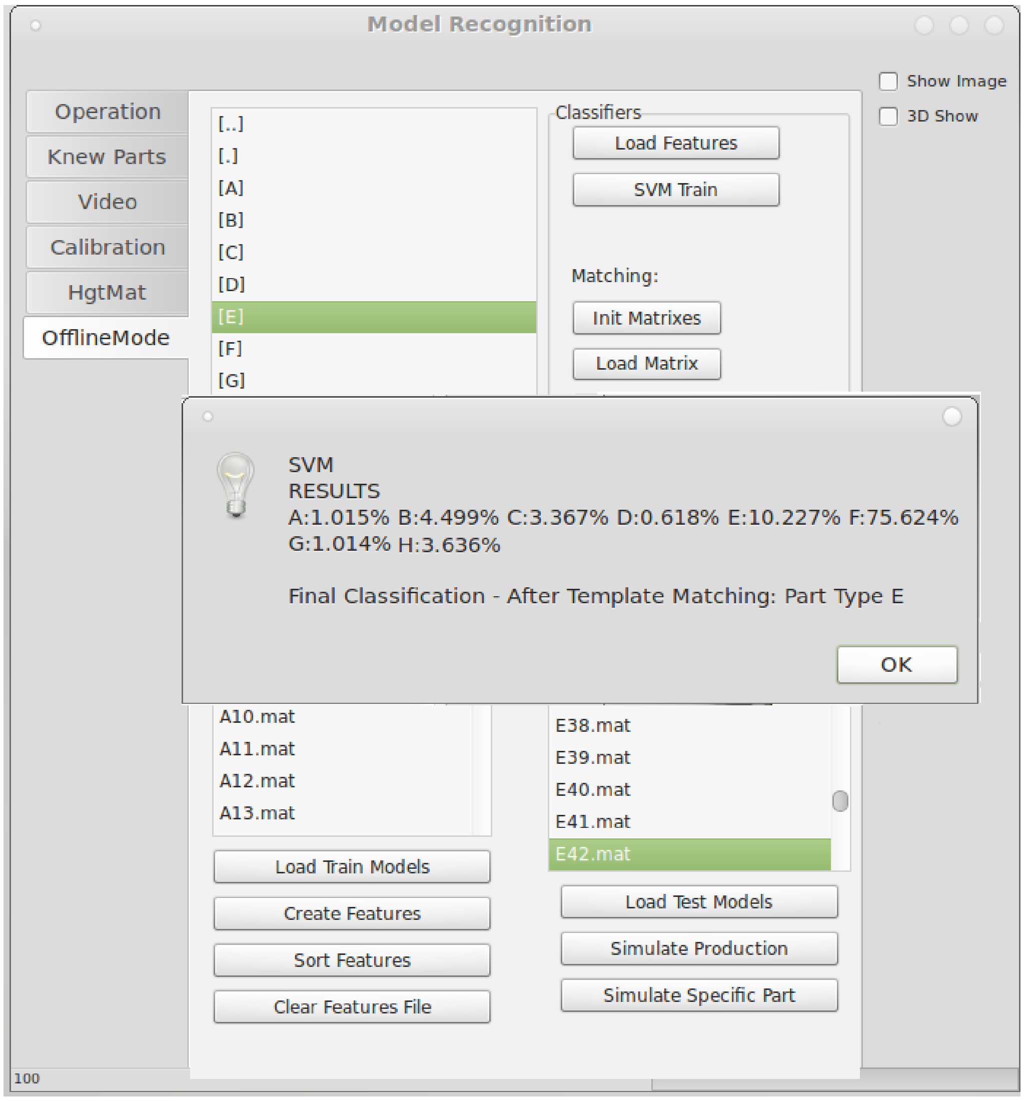
Figure 6.24 shows a bad classification of SVM and the corresponding correction executed by PM. Figure 6.25 shows a case where the SVM classification is close to the two best class hypotheses; hence, the SVM results are not reliable and once again the PM was applied, and the correct classification was achieved. These two cases illustrate perfectly the increase in the classification rate that the PM algorithm brings to the architecture.

This architecture has a maximum computational time of 3 seconds, using $k = 2$ (choosing the 2 best SVM results).

6.5 Pose Estimation

As explained before, PM will be used also as the objects' pose estimator. This information will then be transferred to the industrial manipulator so it can correct its trajectory.

Therefore, it is necessary to validate the PM pose estimation procedure. To perform this study, one type of object was attached to the industrial robot. Then, 5 samples were taken, with:



SVM
RESULTS
A:1.015% B:4.499% C:3.367% D:0.618% E:10.227% F:75.624%
G:1.014% H:3.636%

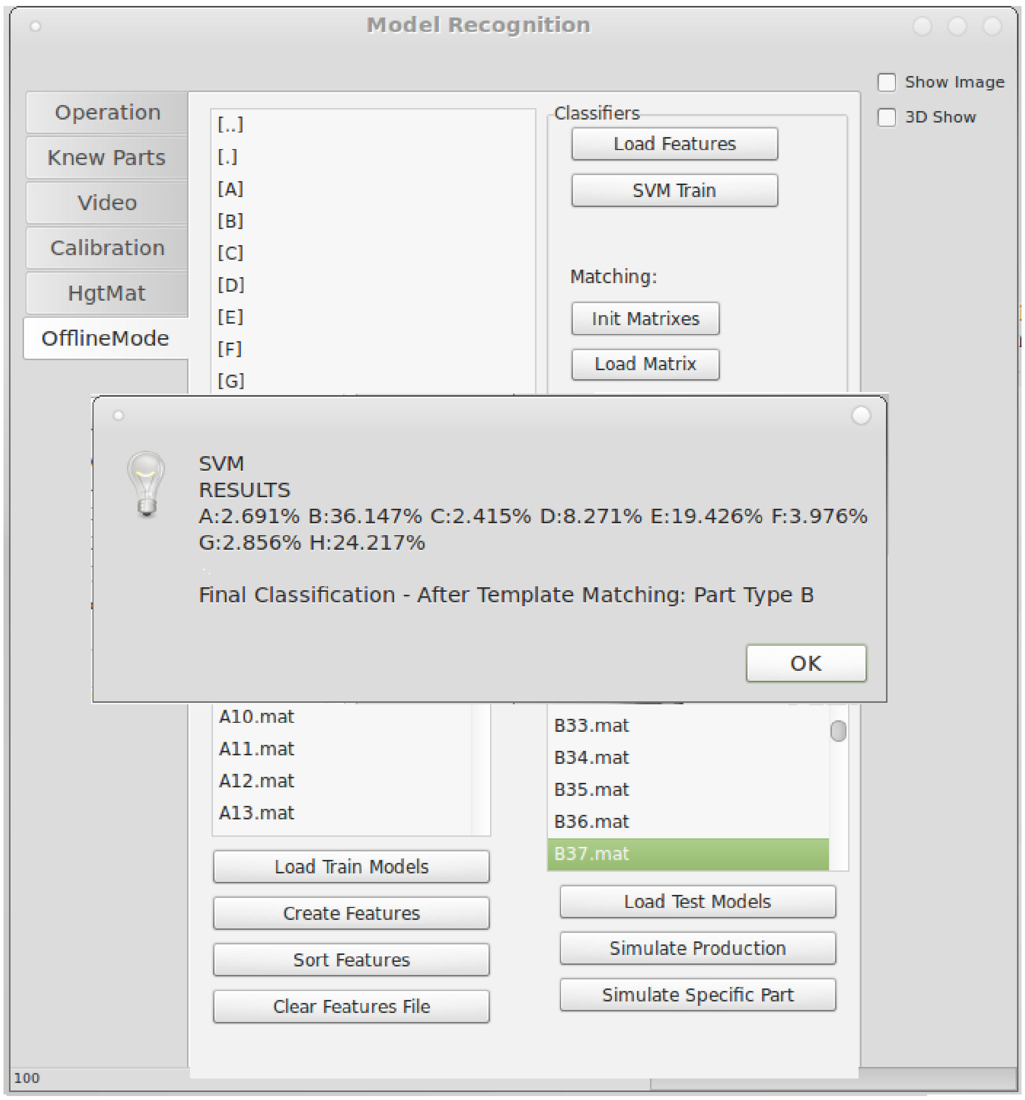
Final Classification - After Template Matching: Part Type E

Figure 6.24: Results of the cascade architecture - Example 1.

- no pose changes with the objective to measure pose estimation repeatability;
- 5 and -5 degree rotation;
- 0.05 m translation along x and y;

The results are presented in the following tables. These results include pose precision maximum absolute error, mean absolute error and standard deviation values.

From Table 6.12 it is possible to conclude that PM pose estimation presents consistent and robust values, considering different models of the same object type with the same position. Presenting a max error of 0.003 m and variance of 0.0015 m.



```
SVM
RESULTS
A:2.691% B:36.147% C:2.415% D:8.271% E:19.426% F:3.976%
G:2.856% H:24.217%

Final Classification - After Template Matching: Part Type B
```

Figure 6.25: Results of the cascade architecture - Example 2.

	x (m)	y (m)	θ (degree)
Max Absolute Error	0.0030	0.0012	0.0572
Mean Absolute Error	0.001	0.0011	0.0114
Standard deviation	0.0015	0.0011	0.0229

Table 6.12: Features Extraction - different observations for the same object.

Tests with a 5 degree object rotation (along the z axis - depth direction) were performed, and new 3D models were captured. These new models were matched to the templates used for the results obtained in table 6.12. The results are shown in Table 6.13.

As it is possible to conclude, the algorithm estimated with high precision the rotation of the object. Note also that a light increase in *x*/*y* max absolute error has occurred. This can have two possible motives: the robot was not completely moving parallel to the 3D

	x (m)	y (m)	θ (degree)
Max Absolute Error	0.0080	0.0027	0.9484
Mean Absolute Error	0.0030	0.0024	0.687549
Standard deviation	0.0037	0.0026	0.1776

Table 6.13: Results for the pose estimation tests, with object 5 degree rotation along θ.

sensor, or the object was not perfectly rotated around its centre (the 3D models mean values were extracted so that they become centred in the world referential). Despite these problems, only a maximum absolute error of 0.94 for the θ estimation and a 0.008 m for the translation axis were obtained.

Next, in Table 6.14 it is presented the results for an object rotation depth direction (z axis). But this time for -5 degrees. The results obtained for pose estimation remain consistent with the previous ones (one more time the mean values of the objects were extracted).

	x (m)	y (m)	θ (degree)
Max Absolute Error	0.0060	0.0060	0.745
Mean Absolute Error	0.0039	0.0026	0.687
Standard deviation	0.0042	0.0026	0.1318

Table 6.14: Results for the pose estimation tests, with object -5 degree rotation along θ.

Finally, the results considering only a translation in object's pose ($\theta = 0$) are presented in Table 6.15. The object was shifted 0.05 along x and y axis (object's width and height).

	x (m)	y (m)	θ (degree)
Max Absolute Error	0.0011	0.0044	0.4755
Mean Absolute Error	0.0009	0.0042	0.4297
Standard deviation	0.0002	0.0002	0.0573

Table 6.15: Results for the pose estimation tests, with object 0.05 m translation along x and y axis (object width).

The PM algorithm,one more time,performs well under the referred conditions. After the presentation of all results, it stays that the geometric matching algorithm detects well and with precision all the displacements verified along all the considered axis. These results leave good expectations for the 6 DOF approach considered for future work. A good initial position estimation must be given so that local minima can be avoided.

6.6 Proposed System vs View Point Feature Histogram

To evaluate the performance of the proposed method in comparison with state-of-the-art solutions, the algorithm proposed by Rusu et. al in [62] and available in the Point Cloud Library [4] was considered.

6.6.1 Object Recognition and Pose Estimation - Point Cloud Library

The Point Cloud Library [65] offers an Object Recognition and 6 DOF Pose Estimation algorithm. It was developed during the last 5 years. It started by the notion of Point Feature Histogram (PFH) and its application for aligning point cloud data views into a consistent global model [63], and for accurately labelling points in a 3D point cloud based on the type of surface the point is lying on [66].

6.6.1.1 Point Feature Histogram

The main principle behind PFH (Point Feature Histogram) is the computation of the relation between a point and its k-Nearest Neighbour. This relation is built using the distance between points and all the relations between their normal vectors. It attempts to capture as best as possible the sampled surface variations by considering all the interactions between the directions of the estimated normals. The result is a highly dimensional hyperspace that provides an informative signature for the feature representation (invariant to the 6D pose). As a prerequisite for the application of Point Feature Histogram, it is necessary to estimate surface normals in a Point Cloud considering a certain amount of neighbours. Furthermore, the computed hyperspace is dependent on the quality of these surface estimation normals.

In Figure 6.26 it is illustrated the computation of PFH for a query point (p_q), considering its k-Nearest Neighbour that are inside a sphere of radius **r**. Therefore, the final PFH descriptor is a feature histogram that contains all the relations between pairs of points in the neighbourhood, and thus has a computational complexity of O(k^2) (k is the number of neighbours). Therefore, considering a point cloud P with n points the computation complexity is $O(nk^2)$.

To compute the relation of two points distances (p_s na p_t) and their respective normals (n_s and n_t) relation it is defined a fixed frame (Darboux frame) with origin at the source point (p_s)(see Figure 6.27 and equations from 6.5 to 6.7).

$$u = n_s \tag{6.5}$$

[4]site: http://pointclouds.org/

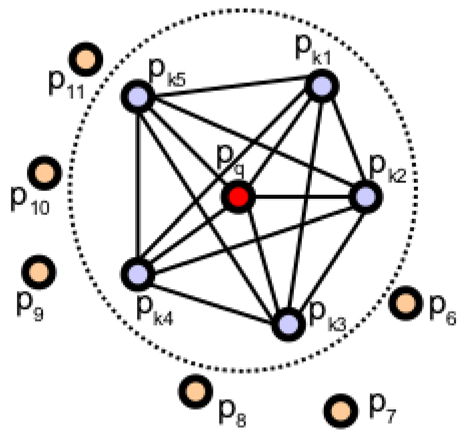


Figure 6.26: Point Feature Histogram - k-Neighbourhood considered points and their relations [62].

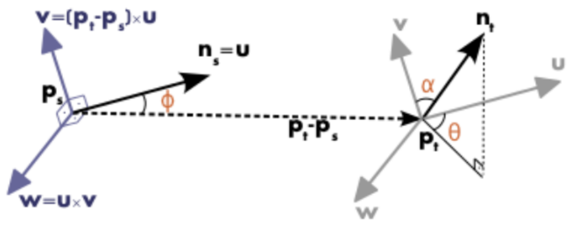


Figure 6.27: The computed Daurbox frame placed at the source frame.

$$v = u \times \frac{(p_t - p_s)}{||p_t - p_s||} \tag{6.6}$$

$$w = u \times v \tag{6.7}$$

Using the above *uvw* frame, the difference between the two normals n_s and n_t can be expressed as a set of angular features as follows:

$$\alpha = v \cdot n_t \tag{6.8}$$

$$\phi = u \cdot \frac{p_t - p_s}{||p_t - p_s||} \tag{6.9}$$

$$\theta = \arctan(\frac{w \cdot n_t}{u \cdot n_t}) \tag{6.10}$$

$$d = ||p_t - p_s|| \tag{6.11}$$

The quadruplet $\langle \alpha, \phi, \theta, d \rangle$ is computed for each pair of two points in k-neighbourhood, therefore, reducing the 12 values (xyz and normal information) of the two points and their normals to 4.

More recently, this approach was optimized for on the fly applications by a drastic reduction of the computational time inherent to PFH - Fast Point Feature Histogram (FPFH) [62].

6.6.1.2 Fast Point Feature Histogram

To achieve this computational time decrease, a modification at the level of mathematical equations and related constraints was performed. With these changes computation complexity was reduced from $O(nk^2)$ to $O(nk)$.

Therefore, to simplify the PFH the authors proceed as follow:

- First step, for each query point p_q it is only computed the relationships between itself and the neighbours. The authors call this the Simplified Point Feature Histogram (SPFH).

- Second step, called FPFH, for each point re-determine k neighbours and use the neighboring SPFH values to weight the final histogram of (p_q) .

$$FPFH(p_q) = SPF(p_q) + \frac{1}{k}\sum_{i=1}^{k}\frac{1}{w_k}.SPF(p_k) \tag{6.12}$$

Where the weight w_k represents a distance between the query point p_q and a neighbour point p_k considering a given metric.

Considering a query point p_q, the first step estimates the SPFH values by creating pairs between itself and its neighbors (illustrated by using red lines in Figure 6.28). This procedure is repeated for all the points in the dataset, followed by a re-weighting of the SPFH values of p_q using the SPFH values of its p_k neighbors, creating the FPFH for p_q.

The extra FPFH connections, resultant due to the additional weighting scheme, are shown with black lines.

Finally, in [64] the authors applied their research work for simultaneous object recognition and pose estimation. For that, they added the viewpoint information in their original Fast Point Feature Histogram, resulting in Viewpoint Feature Histogram (VFH).

6.6.1.3 Viewpoint Feature Histogram

With VFH, the authors' main idea is to estimate the FPFH to the entire object cluster, see Figure 6.29, and to compute additional statistics between the viewpoint direction and the

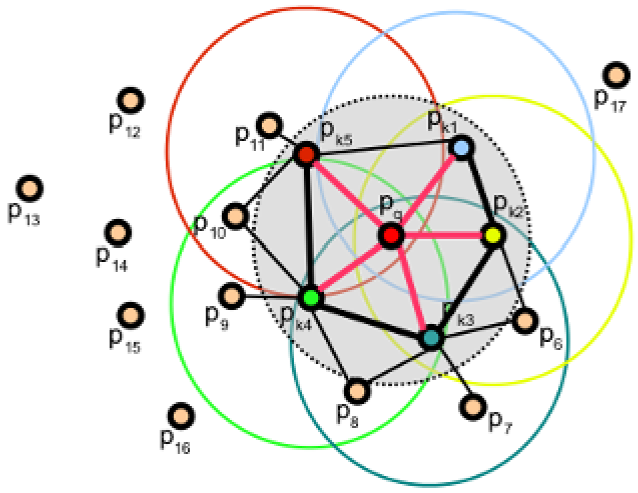


Figure 6.28: Fast Point Feature Histogram - k-Neighbourhood considered points and their relations [62].

normals estimated at each point. Therefore, in their approach the following to descriptors are computed.

- A surface shape component comprised of an extended FPFH, see Figure 6.29;
- A viewpoint direction component, see Figure 6.30;

Figure 6.31 presents this idea with the new feature consisting of two parts.

Adding the viewpoint feature means to add the angle between the view point direction with each point normal. The authors alert that in this operation they do not mean the viewing angle to each normal as it would not be scale invariant, but instead the angle between the central viewpoint direction translated to each normal.

The second component measures the relative pan, tilt and yaw angles as described in FPFH but now measured between the viewpoint direction at the central point and each of the normals on the surface.

The new assembled feature is, therefore, called the Viewpoint Feature Histogram (VFH). The figure below presents this idea with the new feature consisting of two parts:

- a viewpoint direction component.
- a surface shape component comprised of an extended FPFH.

The authors tested this approach for 60 different objects regularly used at home, along 54000 scenes achieving a classification rate of 98.52 %. They also refer that the 6 DOF pose estimation is precise enough for mobile manipulation and grasping. Note that this

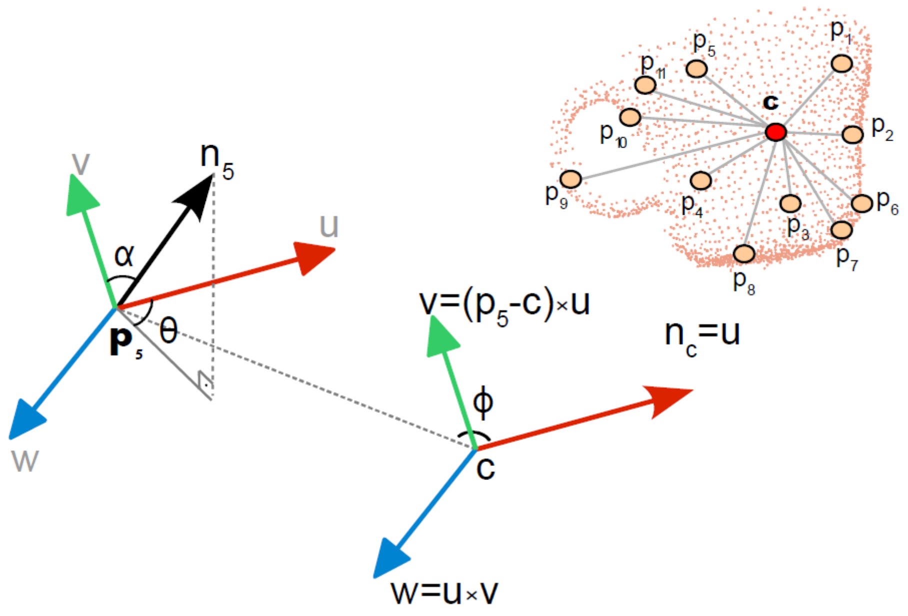


Figure 6.29: The extended Fast Point Feature Histogram collects the statistics of the relative angles between the surface normals at each point to the surface normal at the centroid of the object. The bottom left part of the figure describes the three angular feature for an example pair of points [64].

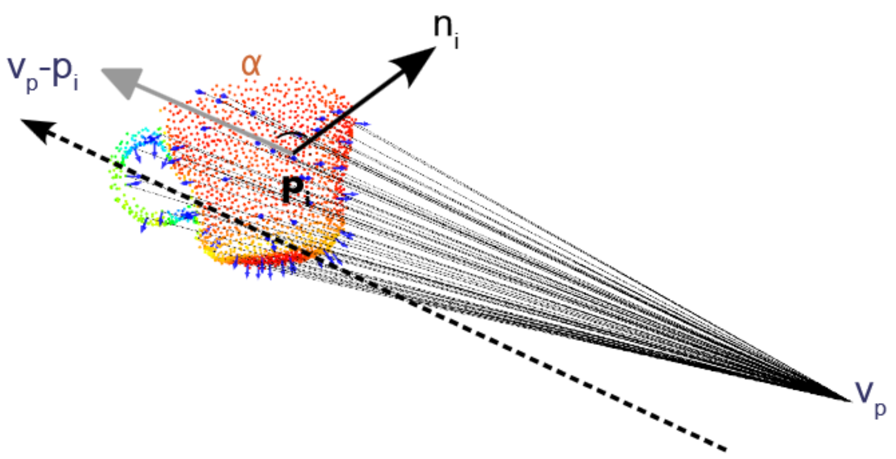


Figure 6.30: The Viewpoint Feature Histogram is created from the extended Fast Point Feature Histogram as seen in Figure 6.28 together with the statistics of the relative angles between each surface normal to the central viewpoint direction [64].

precision is directly related with the number of different viewpoint samples in database (resolution of viewpoint samples in the training phase). This approach is not always possible at industrial environments and also force the use of a large database as well as a position ground truth mechanism in the object teaching phase.

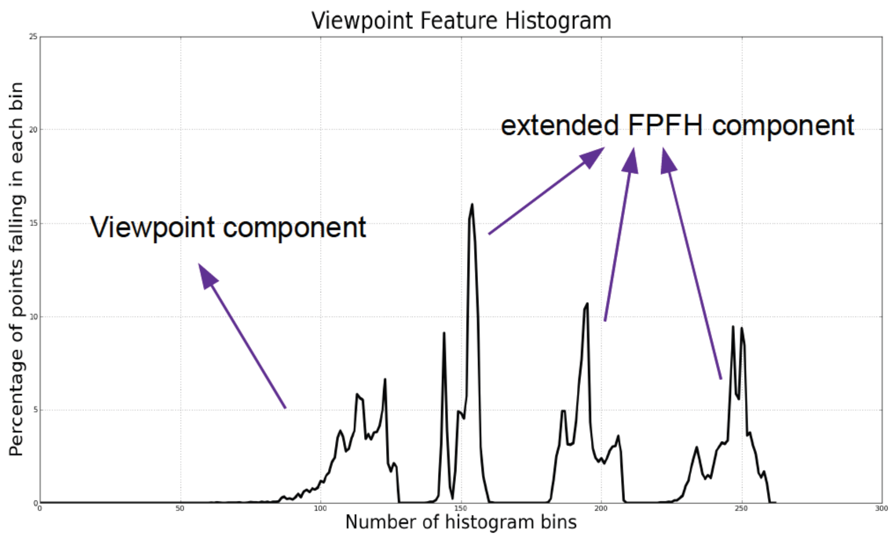


Figure 6.31: An example of the resultant Viewpoint Feature Histogram for one of the objects used. Note the two concatenated components [64].

6.6.2 Comparison Results

Using their approach and the implementation available and developed by PCL authors, the approach presented was tested for the same data set (120 objects from 8 different classes) as the SVM and PM algorithms.

From these 120 samples, their classification procedure was executed achieving a 96.6% classification ratio (considering 4 closest neighbours). For the computation of the normal vectors (features) for each model, it was considered a radius of 0.05 m with a processing time around 4 to 5 seconds.

For the construction of the viewpoint histogram the default values were considered. Its default implementation uses 45 binning for each of the three extended FPFH values, plus 45 binning for distances between each point and the centroid and 128 binning for viewpoint component, which results in a 308-byte array of float values. Measuring the time performance of the this step it was obtained a mean value of 0.0865 seconds.

Finally, for the step of object recognition and having constructed the VFH (View Point Feature Histogram), their algorithm makes use of a fast implementation of k-Nearest Neighbour (search tree). The achieved processing time of this step was about 0.001 seconds considering 16 closest neighbours.

Note that all values were computed not using any special software or hardware set-up.

As it is possible to see and looking at the processing time, the bottleneck of their approach is the computation of the models normal vectors. The time needed for the execution of this step is greater than the time needed for the approach proposed in this thesis. Considering a selection of 2 possible object classes by SVM recognition step and where PM will be applied, all the system procedure takes 3 seconds to perform with standard off the shelf hardware.

Furthermore, the results of classifying the same 120 objects using only to SVM was about 98.3%, and with the cascade approach was about 99.2 %. These results are better than the ones achieved with VFH (96.6 %). The proposed approach showed better response, using a reduced number of training samples as well as to noise measurements. Furthermore, the planar characteristics of some of the objects may have contributed to the lower classification ratio of the VFH based classifier.

Chapter 7

Conclusion and Future Work

7.1 Conclusions

In the last decade and aggravated by the current economic crisis, mass production industries are researching for solutions that allow the enhancement of process efficiency and flexibility, reducing operation costs to become more competitive. If they cannot achieve these goals they are forced to relocate their physical structure to countries with lower production costs, having sever impact in the social-economics state of the leaving country. Therefore, industries must be capable to quickly adapt all the production process and information flow to meet the customization defined by their customers portfolio.

In this scenario, industrial robots are not prepared to work in a high dynamic and flexible production system. Mainly due to their programming procedure and lack of perception skills. These limitations become even more of a problem for SME's where the process information flow is not completely automatized.

Therefore, this thesis focused on the development of an industrial robot complementary and independent system that is responsible to identify and estimate the objects pose that will be target of robot interaction. These mandatory needs are common to production processes where an unstructured transportation system is present, and where the range of operations limit the use of some well know identification system like RFID or bar code.

To solve the present problem, a cascade recognition and pose estimation system was built based on the objects 3D model information. This system is a cascade of two approaches (SVM and PM): SVM - used as a database selector, and Perfect Match - an algorithm for object recognition and pose estimation. The use of SVM is justified by the increase of models in the database and consequently the increase of matchings to perform. This may lead to a processing time slower than the one desired for the production line. Therefore, the idea was to introduce SMVs, that is a much more computationally efficient algorithm considering the amount of classes, to make a pre-selection in the beginning of the classification. The best N (N equal to 2) models candidates are selected from the database and the PM algorithm is applied. The fusion of the information between SVM (object recognition) results and the one retrieved by PM was considered. This approach was tested at an industrial environment, FLUPOL, and a 99.2 % of classification rate was achieved.

All the procedure runs in 3 seconds in an intel core i7 2.93GHz.

As it was possible to understand the pose estimation procedure only estimates 3 DOF. Therefore, objects movement in the other axis are not modelled. It can influence the quality of the PM matching results, both in terms of recognition and pose estimation. Furthermore, a good initial pose estimation for PM is needed. This will minimize the probability of the RPROP algorithm (inerrant of PM approach) to fall in a local minimum.

Considering these limitations some future work is proposed.

7.2 Future Work

As it is possible to conclude, the lack of computation of the 6 DOF, limits the use of the PM algorithm to some scenarios where robot-object interaction is necessary. Therefore, in the future work the main efforts will go in that direction. The first step will be focused in speed up the algorithm computation time, using multi-threading and GPU. With this increase, the extrapolation to 6 DOF will be possible to be made, and having as a basis the results obtained for 3D, good perspectives are seen for the 6 DOF approach. Furthermore, to guarantee a good converge in object geometrical matching, a good initial pose estimation must be given to the algorithm. In this sense, the algorithm proposed in Chapter 7, View Point Feature Histogram, among others will be considered for this purpose.

Although 6 DOF is the main focus, the implementation based in GPU will allow also the increase of robustness in the PM recognition purpose by allowing the consideration of more than one template for each object. Also, it will allow the increase of the number of models in the subset computed by SVM (where the PM will be applied).

Finally, at the level of 3D sensors, the usage of the "sincrovision" [52] concept to the Camera-Laser triangulation sensor will be considered. The idea is to trigger the camera and laser in synchronism, hoping in this way to increase the sensors robustness to environment light and reflections.

7.3 Articles Published or Waitting for Review

Directly from the thesis two articles were published and one is waiting for review:

- Two in international Journals

  [1] Andry Maykol Pinto and Luís F. Rocha and A. Paulo Moreira. Object recognition using laser range finder and machine learning techniques. Robotics and Computer-Integrated Manufacturing. 2013 v. 29, pp 12 - 22.

  [2] Luís F. Rocha, Marcos Ferreira, A. Paulo Moreira and Vitor Santos. Object Recognition and Pose Estimation for Industrial Applications: A Cascade System. Robotics and Computer-Integrated Manufacturing (RCIM). **Submitted and Waiting Review**.

- One in international Conference

  [3] Luís F. Rocha, Marcos Ferreira, Germano Veiga, A. Paulo Moreira and Vitor Santos. Recognizing Industrial Manipulated Parts Using the Perfect Match Algorithm. FAIM 2013, 146-157.

Other articles published during the thesis development:

[4] Marcos Ferreira, Paulo Costa, Luís Rocha, A. Paulo Moreira, Noberto Pires. New Marker for Real-Time Industrial Robot Programming by Motion Imitation. IEEE International Conference on Robotics and Automation (ICRA 2014).

[5] Marcos Ferreira, Paulo Costa, Luís Rocha, Norberto Pires, A. Paulo Moreira. Stereo-based Real-Time 6-DoF Work Tool Tracking for Robot Programming by Demonstration. The International Journal of Advanced Manufacturing Technology. (IJMT 2014).

[6] Marcos Ferreira, Luís Rocha, Paulo Costa, and A. Paulo Moreira. Stereoscopic vision system for human gesture tracking and robot programming by demonstration. FAIM 2013, pp 82-90.

[7] Andry Maykol P., Luís F. Rocha, A. Paulo Moreira and Paulo Costa. Shop Floor Scheduling In a Mobile Robotic Environment, EPIA 2011. 15th Portuguese Conference on Artificial Intelligence Lisbon 2011 - Springer LNCS/LNAI. pp 377-391.

[8] Luís F. Rocha, A. Paulo Moreira, Americo Azevedo. Flexible Internal Logistics based on AGV systems: A case Study. International Conference Management and Control of Production and Logistics (MCPL) Coimbra 2010, pp 248-255.

[9] Luís F. Rocha, A. Paulo Moreira, Americo Azevedo. Increase of Flexibility and Production Rate by an AGV System: A Case Study. International Conference Controlo Coimbra 2010.

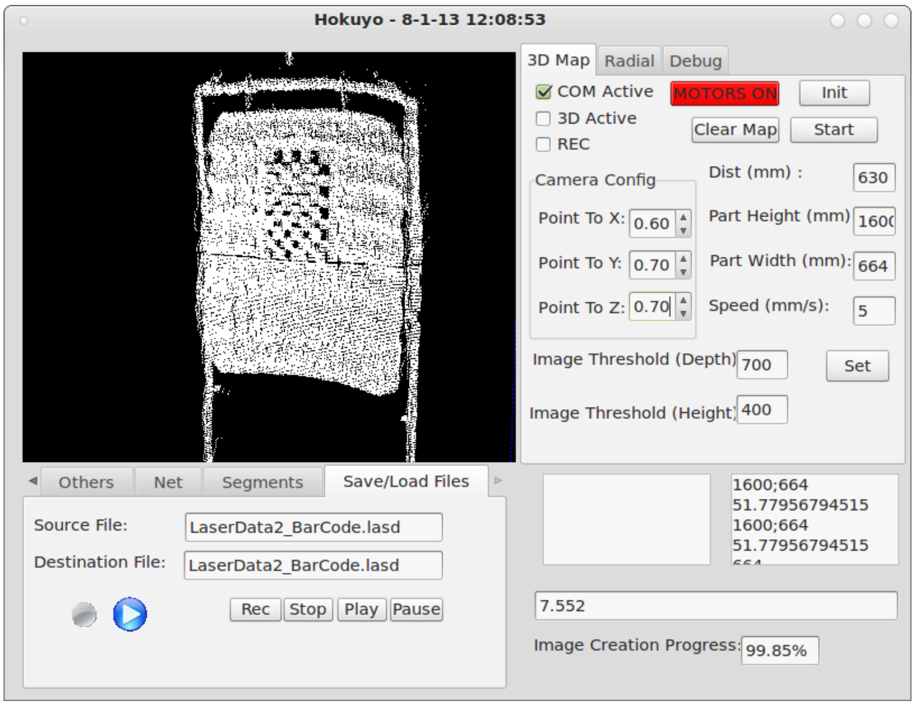
Appendix A

Software Developed for Laser Range Finder Model Extraction

In Figure A.1 is presented the software developed for 3D model extraction using laser range Finder.

When the command of movement is sent, the laser range finder data starts to be recorded and shown in a 3D graphics interface.

Moreover, this software allows also to remotely control the robot using the Ethernet protocol as well as start specific program recorded in robot database.



Hokuyo - 8-1-13 12:08:53

☑ COM Active
☐ 3D Active
☐ REC

Camera Config
Point To X: 0.60
Point To Y: 0.70
Point To Z: 0.70

Dist (mm) : 630
Part Height (mm) 160
Part Width (mm): 664
Speed (mm/s): 5

Image Threshold (Depth) 700
Image Threshold (Height) 400

Source File: LaserData2_BarCode.lasd
Destination File: LaserData2_BarCode.lasd

```
1600;664
51.77956794515
1600;664
51.77956794515
```

7.552

Image Creation Progress: 99.85%

Figure A.1: Sotware developed for Laser Range Finder Object 3D Model Extraction.

Bibliography

[1] D. Acosta, O. Garcia, and J. Aponte. Laser Triangulation for Shape Acquisition in a 3D Scanner Plus Scan. In *Electronics, Robotics and Automotive Mechanics Conference, 2006*, volume 2, pages 14–19, Sept.

[2] M. Antunes, J.P. Barreto, C. Premebida, and U. Nunes. Can stereo vision replace a laser rangefinder? In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5183–5190, 2012.

[3] Franz Aurenhammer. Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure. *ACM Comput. Surv.*, 23(3):345–405, September 1991.

[4] Laura Auria and Rouslan A. Moro. *Support Vector Machines (SVM) as a Technique for Solvency Analysis*. DIW Berlin German Institute for Economic Research, 2008.

[5] S. Barone and A. Bruno. Stereo Vision and Laser Stripers for Three-Dimensional Surface Measurements. *XVI Congreso International de Ingeniería Gráfica*, 4:http://www.egrafica.unizar.es/ingegraf/pdf/Comunicacion17107.pdf, October 2006.

[6] Timothy John Barry and C. Romesh Nagarajah. Object Recognition in Industrial Environments Using Support Vector Machines and Artificial Neural Networks. *The International Journal of Advanced Manufacturing Technology*, 48(5-8):815–821, 2010.

[7] P. Ben-Tzvi, S. Charifa, and M. Shick. Extraction of 3D Images Using Pitch-Actuated 2D Laser Range Finder for Robotic Vision. In *Robotic and Sensors Environments (ROSE), IEEE International Workshop on*, pages 1–6, 2010.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[9] Stefano Carpin, Andreas Birk, and Viktoras Jucikas. On Map Merging. *Robotics and Autonomous Systems*, 53(1):1 – 14, 2005.

[10] V. Cerny. Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm. *Journal of Optimization Theory and applications: vol. 45, No. 1*, 1985.

[11] Jun-Ting Cheng, Cong-Jun Wang, Can Zhao, and Jian-Hua Mo. Design of a Servo Motion System and an Image Sampling and Processing System on a 3D Laser Scanner. *The International Journal of Advanced Manufacturing Technology*, 33:1143–1148, 2007.

[12] J. Clark, G. Zhang, and null. Image Aquisition Using Fixed and Variable Triangulation. In *Image Processing and its Applications, 1995., Fifth International Conference on*, pages 539–543, Jul.

[13] Yan Cui, S. Schuon, D. Chan, S. Thrun, and C. Theobalt. 3D Shape Scanning With a Time-of-Flight Camera. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pages 1173–1180, 2010.

[14] Michele Fenzi, Ralf Dragon, Laura Leal-Taixé, Bodo Rosenhahn, and Jörn Ostermann. 3d Object Recognition and Pose Estimation for Multiple Objects Using Multi-Prioritized RANSAC and Model Updating. In Axel Pinz, Thomas Pock, Horst Bischof, and Franz Leberl, editors, *Pattern Recognition*, volume 7476 of *Lecture Notes in Computer Science*, pages 123–133. Springer Berlin Heidelberg, 2012.

[15] Marcos Ferreira, A. Paulo Moreira, Paulo Malheiros, and Norberto Pires. Highly Flexible Robotized Cells : Application to Small Series Painting. In *20th International Conference Flexible Automation and Intelligent Manufacturing California State University*, Jul 2010.

[16] Marcos Ferreira, António Paulo Moreira, and Pedro Neto. A Low-Cost Laser Scanning Solution for Flexible Robotic Cells: Spray Coating. *The International Journal of Advanced Manufacturing Technology*, 58:1031–1041, 2012.

[17] Jan Flusser, Barbara Zitova, and Tomas Suk. *Moments and Moment Invariants in Pattern Recognition*. Wiley Publishing, 2009.

[18] J.G.D.M. Franca, M.A. Gazziro, A.N. Ide, and J.H. Saito. A 3D Scanning System Based on Laser Triangulation and Variable Field of View. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 1, pages I–425–8, Sept.

[19] Carlos Diogo Pires Carvalho Gomes. Sistema de Classificação de Tecidos para a Indústria Automóvel, uma Abordagem Baseada em Aprendizagem. Master's thesis, Faculdade de Ciências e tecnologia Universidade de Coimbra, 2012.

[20] Alexandre Reis Graeml and João Mário Csillag. Customization in the Manufactirng Industry: Survey Results in Southeaster Brazil. *Journal of Information Systems and Technology Managment*, 6(3):p395–412, 2010.

[21] Mohamad H. Hassoun. *Fundamentals of Artificial Neural Networks* . MIT Press, 1995.

[22] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, corrected edition, August 2003.

[23] Jose-Juan Hernandez-Lopez, Ana-Linnet Quintanilla-Olvera, José-Luis Lopez-Ramirez, Francisco-Javier Rangel-Butanda, Mario-Alberto Ibarra-Manzano, and Dora-Luz Almanza-Ojeda. Detecting Objects Using Color and Depth Segmentation With Kinect Sensor. *Procedia Technology*, 3(0):196 – 204, 2012. The 2012 Iberoamerican Conference on Electronics Engineering and Computer Science.

[24] Low Kin Huat. *Industrial Robotics: Programming, Simulation and Applications*. Pro Literatur Verlag, Germany / ARS, Austria, 2006.

[25] M. Peña-Cabrera* I. Lopez-Juarez and A.V. Reyes-Acosta. *Using Object's Contour and Form to Embed Recognition Capability into Industrial Robots*. InTech., University Campus STeP Ri, Croatia, 2010.

[26] James R. Janesick. *Scientific Charge-Coupled Devices*. SPIE Press Book, 2001.

[27] Yong Jiang, Ning Xi, Qin Zhang, and Yunyi Jia. Target Object Identification and Localization in Mobile Manipulations. In *Robotics and Biomimetics (ROBIO), IEEE International Conference on*, pages 144–149, 2011.

[28] M. A. Khabou, L. Hermi, and M. B. H. Rhouma. Shape Recognition Using Eigenvalues of the Dirichlet Laplacian. *Pattern Recogn.*, 40(1):141–153, January 2007.

[29] Kyekyung Kim, Joongbae Kim, Sangseung Kang, Jaehong Kim, and Jaeyeon Lee. Object recognition For Cell Manufacturing SYstem. In *Ubiquitous Robots and Ambient Intelligence (URAI), 2012 9th International Conference on*, pages 512–514, 2012.

[30] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[31] D. Klimentjew, M. Arli, and Jianwei Zhang. 3D Scene Reconstruction Based on a Moving 2D Laser Range Finder for Service-Rfobots. In *Robotics and Biomimetics (ROBIO), IEEE International Conference on*, pages 1129–1134, 2009.

[32] D. Klimentjew, N. Hendrich, and Jianwei Zhang. Multi Sensor Fusion of Camera and 3D Laser Range Finder for Object Recognition. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), IEEE Conference on*, pages 236–241, 2010.

[33] R. Koker, C. Oz, and A. Ferikoglu. Development of a Vision Based Object Classification System for an Industrial Robotic Manipulator. In *Electronics, Circuits and Systems. ICECS 2001. The 8th IEEE International Conference on*, volume 3, pages 1281–1284 vol.3, 2001.

[34] Martin Lauer, Sascha Lange, and Martin Riedmiller. Calculating the Perfect Match: an Efficient and Accurate Approach for Robot Self-Localization. In *RoboCup 2005: Robot Soccer World Cup IX, Springer, Lecture Notes in Computer Science*, volume 4020, 2006.

[35] J. Le Moigne and A.M. Waxman. Multi-Resolution Grid Patterns for Building Range Maps. In *Proc. Vision Conf.*, volume 8, pages 22–39, 1985.

[36] J.R.J. Lee, M.L. Smith, and L.N. Smith. A New Approach to the Three-Dimensional Quantification of Angularity Using Image Analysis of the Size and Form of Coarse Aggregates. *Engineering Geology*, 91(2-4):254 – 264, 2007.

[37] Wang Lei, Bo Mei, Gao Jun, and Ou ChunSheng. A Novel Double Triangulation 3D Camera Design. In *Information Acquisition, 2006 IEEE International Conference on*, pages 877–882, Aug.

[38] R.K. Lenz and R.Y. Tsai. Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 68–75, Mar.

[39] Jingchao Li, Zhenjiang Miao, Xiangqian Liu, and Yanli Wan. 3D Reconstruction based on stereovision and texture Mapping. *Paparoditis N., Pierrot-Deseilligny M., Mallet C., Tournaire O. (Eds), IAPRS, Vol. XXXVIII, Part 3B - Saint-Mande, France, September 1-3, 2010*, 2010.

[40] Yun Liu, Yanmin Yin, and Shujun Zhang. Hand Gesture Recognition Based on HU Moments in Interaction of Virtual Reality. In *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2012 4th International Conference on*, volume 1, pages 145–148, 2012.

[41] O. Ludwig, C. Premebida, U. Nunes, and R. Araujo. Evaluation of boosting-svm and srm-svm cascade classifiers in laser and vision-based pedestrian detection. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1574–1579, 2011.

[42] M. Maruyama and S. Abe. Range Sensing by Projecting Multi-Slits with Random Cuts. In *Industrial Applications of Machine Intelligence and Vision, 1989., International Workshop on*, pages 163–168, 1989.

[43] M. Masaeli, G. Fung, and Jennifer G. Dy. From Transformation-Based Dimensionality Reduction to Feature Selection. *27th International Conference on Machine Learning, Haifa, Israel*, 2010.

[44] Yang Mingqiang, Kpalma Kidiyo, and Ronsin Joseph. A Survey of Shape Feature Extraction Techniques. *Pattern Recognition*, pages 43–90, 2008.

[45] H. Morita, K. Yajima, and S. Sakata. Reconstruction of surfaces of 3-d objects by m-array pattern projection method. In *Computer Vision., Second International Conference on*, pages 468–473, 1988.

[46] Sergiu Nedevschi, Radu Danescu, Dan Frentiu, Tiberiu Marita, Florin Oniga, Ciprian Pocol, Thorsten Graf, and Rolf Schmidt. High Accuracy Stereovision Approach for Obstacle Detection on Non-Planar Roads. In *in IEEE Inteligent Engineering Systems (INES*, pages 211–216, 2004.

[47] Matthias Nieuwenhuisen, David Droeschel, Dirk Holz, Joerg Stückler, Alexander Berner, Jun Li, Reinhard Klein, and Sven Behnke. Mobile Bin Picking with an Anthropomorphic Service Robot. *Accepted for IEEE International Conference on Robotics and Automation (ICRA), to appear*, may 2013.

[48] M. Oliveira and V. Santos. Combining View-based Object Recognition with Template Matching for the Identification and Tracking of Fully Dynamic Targets. *7th Conference on Mobile Robots and Competitions*, 2007.

[49] M. Oliveira and V. Santos. Automatic Detection of Cars in Real Roads using Haar-like Features. *8th Conference on Automatic Control*, July 2008.

[50] H. Pan and L. Z. Xia. Efficient Object Recognition Using Boundary Representation and Wavelet Neural Network. *Trans. Neur. Netw.*, 19(12):2132–2149, December 2008.

[51] S. Paschalakis and P. Lee. Pattern Recognition in Grey Level Images Using Moment Based Invariant Features. In *Image Processing And Its Applications. Seventh International Conference on (Conf. Publ. No. 465)*, volume 1, pages 245–249 vol.1, 1999.

[52] A. Paulo Moreira Paulo Malheiros, Paulo Costa and José Carlos Lopes. Real-time Teaching of Industrial Robots Using a Synchronised Stereoscopic Vision System. In *ROBOTICA 2009 - 9th Conference on Mobile Robots and Competitions*, pages 41–45, 7th May 2009.

[53] M. Pena, I. Lopez, and R. Osorio. Invariant Object Recognition Robot Vision System for Assembly. In *Electronics, Robotics and Automotive Mechanics Conference, 2006*, volume 1, pages 30–36, 2006.

[54] Andry Maykol Pinto, Luís F. Rocha, and A. Paulo Moreira. Object Recognition Using Laser Range Finder and Machine Learning Techniques. *Robotics and Computer-Integrated Manufacturing*, 29(1):12 – 22, 2013.

[55] Miguel Pinto, A. Paulo Moreira, Paulo Costa, Marcos Ferreira, and Paulo Malheiros. Robotic manipulator and artificial vision system for picking cork pieces in a conveyor belt. In *20th International Conference Flexible Automation and Intelligent Manufacturing California State University*, Jul 2010.

[56] Miguel Pinto, António Paulo Moreira, and Aníbal Matos. *Robots Localisation in Indoor and Service Scenarios: Two Approaches: Landmark-Based and Three-Dimensional Map-Based.* Lambert Academic Publishing, 2013.

[57] M. Pontil and A. Verri. Support Vector Machines for 3D Object Recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(6):637–646, 1998.

[58] C. Premebida, O. Ludwig, M. Silva, and U. Nunes. A cascade classifier applied in pedestrian detection using laser and image-based features. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 1153–1159, 2010.

[59] C Rath. Self-Localization of a Biped Robot in the RoboCup Domain. Master's thesis, Institute for Software Technology, Graz University of Technology, 2010.

[60] Luís F. Rocha, Marcos Ferreira, Germano Veiga, A. Paulo Moreira, and Vitor Santos. Recognizing Industrial Manipulated Parts Using the Perfect Match Algorithm. *FAIM*, 2013.

[61] D.N. Rockmore. The FFT: An Algorithm the Whole Family Can Use. *Computing in Science Engineering*, 2(1):60 –64, jan/feb 2000.

[62] R.B. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3212–3217, 2009.

[63] R.B. Rusu, N. Blodow, Z.C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3384–3391, 2008.

[64] R.B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the Viewpoint Feature Histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162, 2010.

[65] R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1–4, 2011.

[66] R.B. Rusu, Z.C. Marton, N. Blodow, and M. Beetz. Learning informative point classes for the acquisition of object model maps. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, pages 643–650, 2008.

[67] Giovanna Sansoni and Elisa Redaelli. A 3D Vision System Based on One-Shot Projection and Phase Demodulation for Fast Profilometry. *Measurement Science and Technology*, 16(5):1109, 2005.

[68] Giovanna Sansoni, Marco Trebeschi, and Franco Docchio. State-of-the-art and applications of 3D imaging sensors in industry, cultural heritage, medicine, and criminal investigation. *Sensors*, 9:568–601, January 2009.

[69] J Santolaria, J J Pastor, F J Brosed, and J J Aguilar. A one-step intrinsic and extrinsic calibration method for laser line scanner operation in coordinate measuring machines. *Measurement Science and Technology*, 20(4), 2009.

[70] Dieter Schmoeckel. Developments in automation, flexibilization and control of forming machinery. *{CIRP} Annals - Manufacturing Technology*, 40(2):615 – 622, 1991.

[71] S. Schuon, C. Theobalt, J. Davis, and S. Thrun. High-Quality Scanning Using Time-of-Flight Depth Superresolution. In *Computer Vision and Pattern Recognition Workshops. CVPRW '08. IEEE Computer Society Conference on*, pages 1–7, 2008.

[72] Dejan Seatovic, Rolf Grüninger, Anken Thomas, and Holpp Martin. 3D Object Recognition, Localization and Treatment of Rumex Obtusifolius in its Natural Environment. In *1st International Conference on Machine Control & Guidance*, 2008.

[73] Sebastian Seung. *Introduction to Neural Networks Lecture 1*. The Seung Lab - MIT, 2002.

[74] Ajay Singholi, Deepti Chhabra, and Mohammad Ali. Towards improving the performance of flexible manufacturing sytem: a case study. *Journal of Enginneering and Management*, 3(1):p87–115, 2010.

[75] Yasushi Sumi and Fumiaki Tomita. 3D Object Recognition Using Segment-Based Stereo Vision. In *In Proc. of ACCV'98, II*, pages 249–256, 1998.

[76] R.Y. Tsai. A versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *Robotics and Automation, IEEE Journal of*, 3(4):323–344, August.

[77] A. Wahi, P. Ravi, and 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. In *Computing Communication and Networking Technologies (ICCCNT), 2010 International Conference on*, pages 1–6, 2010.

[78] Guo-Qing Wei and G. Hirzinger. Active self-calibration of hand-mounted laser range finders. *Robotics and Automation, IEEE Transactions on*, 14(3):493–497, 1998.

[79] Thomas Whelan, Sonja Stüdli, John McDonald, and RichardH. Middleton. Efficient localization for robot soccer using pattern matching. In Reiner Hähnle, Jens Knoop, Tiziana Margaria, Dietmar Schreiner, and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification, and Validation*, Communications in Computer and Information Science, pages 16–30. Springer Berlin Heidelberg, 2012.

[80] Manuel Wopfner, Jonas Brich, Siegfried Hochdorfer, and Christian Schlegel. Mobile Manipulation in Service Robotics: Scene and Object Recognition with Manipulator-Mounted Laser Ranger. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 1–7, 2010.

[81] Ke-Ru Wu, An Yan, Juan yu Liu, Dong Zhang, and Wu Yao. Reconstruction and Analysis of 3-D Profile of Fracture Surface of Concrete. *Cement and Concrete Research*, 30(6):981 – 987, 2000.

[82] J. Zavadil, V. Santos, and J. Tuma. Traffic lights recognition for robotic competition. *Inproceedings of MATLAB conference*, 2011.

[83] Hao Zhang, A.C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In *Computer Vision and*

Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 2126–2136, 2006.

[84] Zheng Zhao. Advancing Feature Selection Research. Technical report, arizona state university, 2010.