



Universitat d'Alacant
Universidad de Alicante

Contributions to 3D object
recognition and social robotics

Félix Escalona Moncholí



Tesis **Doctorales**

UNIVERSIDAD de ALICANTE

Unitat de Digitalització UA

Unidad de Digitalización UA



Universitat d'Alacant
Universidad de Alicante

Instituto Universitario de Investigación Informática

Contributions to 3D data processing and social robotics

Félix Escalona Moncholí

TESIS PRESENTADA PARA ASPIRAR AL GRADO DE
DOCTOR POR LA UNIVERSIDAD DE ALICANTE

MENCIÓN DE DOCTOR INTERNACIONAL

PROGRAMA DE DOCTORADO EN INFORMÁTICA

Dirigida por:

Dr. Miguel Ángel Cazorla Quevedo

Dr. Diego Viejo Hernando

2021

The thesis presented in this document has been reviewed and approved
for the
INTERNATIONAL PhD HONOURABLE MENTION

I would like to thank the advises and contributions for this thesis of the
external reviewers:

Professor Robert Fisher
(University of Edinburgh)

This thesis is licensed under a CC BY-NC-SA International License (Creative Commons AttributionNonCommercial-ShareAlike 4.0 International License). You are free to share — copy and redistribute the material in any medium or format Adapt — remix, transform, and build upon the material. The licensor cannot revoke these freedoms as long as you follow the license terms. Under the following terms: Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. NonCommercial — You may not use the material for commercial purposes. ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

Document by Félix Escalona Moncholí.

Universitat d'Alacant
Universidad de Alicante



*A mi familia,
Reme, Laura y Félix,
en su memoria.*



Universitat d'Alacant
Universidad de Alicante

*El trabajo duro es un talento. La capacidad de seguir
intentándolo cuando otros abandonan es un talento.*

Garry Kasparov



Universitat d'Alacant
Universidad de Alicante

Agradecimientos

Llegados al término de este viaje, merece la pena hacer un alto en el camino para agradecer el apoyo recibido, que me ha dado fuerzas en los peores momentos para llegar hasta donde estoy hoy. Perdonad por no mencionar a ninguno directamente, pero es que me gusta poner nombres ya que siempre se comete el gran pecado de olvidar mencionar a alguien. No obstante, estoy seguro de que sabréis que me he acordado de vosotros y lo mucho que os agradezco vuestra ayuda.

Gracias a mi familia, os lo debo todo. Hemos pasado muy malos momentos, pero los hemos superado juntos y nos ha hecho más fuertes. Sé que Papá nos ha ayudado allá desde donde esté.

A mis amigos del colegio. Prácticamente llevamos toda la vida juntos y, aunque haya habido momentos en los que no nos hemos visto tanto como me gustaría, sé que siempre puedo contar con vosotros.

A mis amigos del instituto. Me acogisteis en una etapa de cambios, el paso del colegio al instituto, y me ayudasteis a adaptarme rápidamente y sentir como si hubiese estado toda la vida allí. Aunque en estos tiempos locos que estamos viviendo, con pandemia mundial incluida, apenas nos hemos visto, habéis sido un gran apoyo para mí en estos últimos años.

A mis compañeros del laboratorio. Desde el primer momento que llegué me acogisteis como a uno más y me hicisteis sentir parte de esa gran familia, no tanto en número como en sentimiento, que somos. Hemos compartido viajes, congresos, artículos y muchísimas horas de frustración por el trabajo que no sale, pero habéis sabido siempre sacarme una sonrisa y darme ánimos para seguirlo intentando.

A mis tutores. Tenéis mucha culpa de que yo hoy esté presentando esta

tesis. Me habéis iluminado el camino siempre que los resultados no eran los esperados y me habéis dado la confianza necesaria para explorar y salir de mi zona de confort.

Muchas gracias a todos por vuestro apoyo. Dicen que Dios no castiga dos veces así que, por haberme aguantado tantos años, os absuelve de tener que leeros lo que viene a continuación. Si decidís continuar, será solamente bajo vuestra responsabilidad ;).

Alicante, 4 de mayo de 2021

Félix Escalona Moncholí



Universitat d'Alacant
Universidad de Alicante

Resumen

En esta tesis se realiza un estudio de la inteligencia artificial aplicada a los datos 3D y a la robótica social.

La primera parte del documento está dedicada al reconocimiento de objetos en 3D. El reconocimiento de objetos consiste en la detección y categorización automática de los objetos que aparecen en una escena. Esta capacidad es una necesidad importante para los robots sociales, ya que les permite entender e interactuar con su entorno. Los métodos basados en imágenes han sido ampliamente estudiados con grandes resultados, pero sólo se basan en características visuales y pueden confundir diferentes objetos con apariencias similares, por lo que los datos 3D pueden ayudar a mejorar estos sistemas utilizando características topológicas. En esta parte, presentamos diferentes técnicas que utilizan datos 3D puros.

La segunda parte de la tesis trata sobre el mapeo del entorno. El mapeo del entorno consiste en construir un mapa que pueda ser utilizado por un robot para ubicarse. Esta capacidad les permite llevar a cabo una estrategia de navegación más elaborada, que es tremendamente utilizable por un robot social para interactuar con las diferentes habitaciones de una casa y sus objetos. En esta sección, exploraremos los mapas 2D y 3D y su perfeccionamiento con el reconocimiento de objetos.

Por último, la tercera parte de este trabajo trata sobre la robótica social. La robótica social se centra en servir a las personas en una interacción afectiva más que en realizar una tarea mecánica. Las secciones anteriores están relacionadas con dos capacidades principales de un robot social, y esta sección final contiene una revisión sobre este tipo de robots y otros proyectos que exploran otros aspectos de los mismos.

Abstract

In this thesis, a study of artificial intelligence applied to 3D data and social robotics is carried out.

The first part of the present document is dedicated to 3D object recognition. Object recognition consists on the automatic detection and categorisation of the objects that appear in a scene. This capability is an important need for social robots, as it allows them to understand and interact with their environment. Image-based methods have been largely studied with great results, but they only rely on visual features and can confuse different objects with similar appearances (picture with the object depicted in it), so 3D data can help to improve these systems using topological features. For this part, we present different novel techniques that use pure 3D data.

The second part of the thesis is about the mapping of the environment. Mapping of the environment consists on constructing a map that can be used by a robot to locate itself. This capability enables them to perform a more elaborated navigation strategy, which is tremendously usable by a social robot to interact with the different rooms of a house and its objects. In this section, we will explore 2D and 3D maps and their refinement with object recognition.

Finally, the third part of this work is about social robotics. Social robotics is focused on serving people in a caring interaction rather than to perform a mechanical task. Previous sections are related to two main capabilities of a social robot, and this final section contains a survey about this kind of robots and other projects that explore other aspects of them.

Contents

List of Figures	xxiii
List of Tables	xxxix
List of Algorithms	xxxix
1 Introduction	1
1.1 Introduction and Motivation	1
1.2 Methodology	4
1.2.1 ASUS Xtion Pro Live	4
1.2.2 Pepper Robot	5
1.2.2.1 The issue with the 3D sensor of the Pepper robot	11
1.2.3 ROS	12
1.2.4 Gazebo	12
1.2.5 Gmapping	13
1.2.6 Caffe	14
1.2.7 Tensorflow and Keras	15
1.2.8 Darknet	15
1.2.9 PCL	15
1.2.10 Datasets for 3D recognition	16
1.2.10.1 Princeton ModelNet	17
1.2.11 Datasets for action recognition	19
1.2.11.1 KARD	20
1.3 Proposal and goals	22

1.4	Structure of the dissertation	23
2	Background	25
2.1	Introduction	25
2.2	2D Object recognition	26
2.2.1	Model-based	27
2.2.2	Appearance-based	29
2.2.3	Deep Learning-based	31
2.3	3D Object recognition	33
2.4	Fractals used for classification purposes	37
2.5	3D Mapping	38
2.5.1	Visual-inertial SLAM	38
2.5.2	Registration of a scene	40
2.6	Social robotics for personal assistance	45
2.7	Social robotics for rehabilitation	48
2.8	People re-identification	52
3	3D Object Recognition	55
3.1	Introduction	55
3.2	Object recognition using free form surfaces	56
3.2.1	Approach	56
3.2.2	Nurbs surfaces	57
3.2.2.1	Nurbs fitting	58
3.2.2.2	Similarity metric between two Nurbs surfaces	58
3.2.3	Network architecture	61
3.2.3.1	Nurbs layer	61
3.2.3.2	Voxelization layer	62
3.2.3.3	Fully connected layer	64
3.2.4	Experiments	64
3.2.4.1	ModelNet10	64
3.2.4.2	ModelNet40	66
3.2.5	Conclusions and future works	67
3.3	Object recognition using fractals	70
3.3.1	Proposal	70
3.3.1.1	3D fractal dimension algorithm	71

3.3.1.2	3D voxelized fractal descriptor	72
3.3.2	Experimentation	76
3.3.2.1	Results on ModelNet10 using the 1D descriptor	76
3.3.2.2	Results on ModelNet10 using VFD	77
3.3.2.3	Results on ModelNet40 using VFD	80
3.3.3	Conclusions and future Work	82
4	Mapping of the environment	87
4.1	Introduction	87
4.2	3D object mapping using a labelling system	88
4.2.1	Semantic labelling	89
4.2.2	3D mapping of the environment	90
4.2.2.1	Registration by odometry	90
4.2.2.2	Visual registration	91
4.2.3	Global image annotation using CNN	92
4.2.4	3D mapping with semantic labels	94
4.2.5	Experiments and results	98
4.2.5.1	Object recognition over robotic platform	98
4.2.5.2	Recognition of multiple object instances of the same class	100
4.2.6	Conclusion	102
4.3	Enhanced Ambient Assisted Living with a social robot	105
4.3.1	System description	105
4.3.2	Using a robot for dangerous areas detection	106
4.3.2.1	Detection of objects on the ground	107
4.3.2.2	Superficial object detector	109
4.3.3	3D world references registration between AAL and the robot	110
4.3.4	Experimentation	112
4.3.4.1	Objects on the ground tracker algorithm experiments	113
4.3.4.2	Superficial object detector algorithm experiments in office environment	115

4.3.4.3	Superficial object detector algorithm experiments in home environment	117
4.3.5	Conclusions	118
4.4	Mapping, navigation and planning system for a social robot with semantic localization	120
4.4.1	Semantic localization system	120
4.4.2	Motion planning system	122
4.4.3	Navigation and mapping	125
4.4.4	Experimentation, results and discussion	127
4.4.4.1	Motion planning system experimentation	128
4.4.4.2	Navigation and mapping experimentation	130
4.4.4.3	Conclusions and limitations	133
4.5	Fusion of Pepper robot and estimated depth maps method for improved 3D perception	135
4.5.1	The issue with the 3D sensor of the Pepper robot	135
4.5.2	Other issues related to 3D cameras	136
4.5.3	Issues related to monocular depth estimation	137
4.5.4	FusionV1: Fusing the output of a depth estimation from monocular frames system and the Pepper depth maps	138
4.5.5	Issues related to the previous fusion method	140
4.5.6	FusionV2: Refining the fusion of monocular and Pepper depth maps	140
4.5.7	Experimentation	142
4.5.8	Conclusions	146
5	Investigations applied to social robotics	149
5.1	Introduction	149
5.2	Socially assistive robots for older adults and people with autism	150
5.2.1	Socially Assistive Robots	151
5.2.2	Older adult care	152
5.2.3	Training communication and social interaction in children with autism	157

5.2.4	Conclusions	164
5.3	EVA: Evaluating at-home rehabilitation exercises using augmented reality and low-cost sensors	166
5.3.1	Proposal	167
5.3.1.1	Common 3D coordinate frame estimation	169
5.3.1.2	Human 3D pose estimation from monocular frames	170
5.3.1.3	Scoring the user's performance	173
5.3.2	Augmented reality application	177
5.3.2.1	User interface	178
5.3.3	Rehabilitation exercises	180
5.3.4	Experimentation and results	181
5.3.4.1	Human 3D pose estimation experiments	181
5.3.4.2	DTW scoring experiments	184
5.3.5	Conclusions and future work	184
5.4	Semantic visual recognition in a cognitive architecture for social robots	187
5.4.1	Person identification memory system	188
5.4.2	Experimentation	190
5.4.2.1	Dataset	191
5.4.2.2	RoboCup challenge	191
5.4.2.3	Person identification system experiments	192
5.4.2.4	Limitations	200
5.4.3	Conclusion and future works	202
6	Conclusions	203
6.1	Conclusions	203
6.2	Contributions of the thesis	205
6.3	Publications	205
6.4	Future work	208
	Appendices	211
A	Introducción	213
A.1	Introducción y motivación	213

A.2	Metodología	216
A.2.1	ASUS Xtion Pro Live	216
A.2.2	Robot Pepper	217
A.2.2.1	El problema del sensor 3D del robot Pepper	223
A.2.3	ROS	224
A.2.4	Gazebo	224
A.2.5	Gmapping	225
A.2.6	Caffe	226
A.2.7	Tensorflow y Keras	227
A.2.8	Darknet	227
A.2.9	PCL	227
A.2.10	Conjuntos de datos para reconocimiento 3D	228
A.2.10.1	Princeton ModelNet	229
A.2.11	Conjuntos de datos para el reconocimiento de acciones	232
A.2.11.1	KARD	233
A.3	Propuesta y objetivos	235
A.4	Estructura de la tesis	236
B	Conclusiones	239
B.1	Conclusiones	239
B.2	Aportaciones de la tesis	241
B.3	Publicaciones	242
B.4	Trabajos futuros	245
	Bibliography	247

List of Figures

1.1	ASUS Xtion Pro Live Sensor	4
1.2	Point cloud from Xtion	4
1.3	Pepper Robot	6
1.4	Dimensions of the Pepper Robot	6
1.5	Pepper sensors and actuators	7
1.6	Joints of the Robot Pepper	8
1.7	Sign convention for Robot Pepper coordinate systems	8
1.8	Turning range of the Robot Pepper's head	8
1.9	Turning range of the left limb of Robot Pepper	9
1.10	Robot Pepper's right limb rotation range	9
1.11	Robot Pepper's hip swing range	9
1.12	Robot Pepper's hand	10
1.13	Diagram of the Robot Pepper base	10
1.14	Pepper and standalone Xtion captures	12
1.15	Example of Gazebo simulation	13
1.16	Gmapping example	14
1.17	PCL display example	16
1.18	Comparative between instances of <code>night_stand</code> and <code>dresser</code> in frontal and side view	18
1.19	Comparative between instances of <code>cup</code> and <code>vase</code> in frontal and top view.	19
1.20	Comparative between instances of <code>plant</code> and <code>flower_pot</code> in frontal and top view.	19
1.21	Image examples from KARD	21

2.1	SLAM Scheme	40
2.2	SLAM process (1)	40
2.3	SLAM process (2)	41
2.4	SLAM process (3)	41
2.5	SLAM process (4)	41
2.6	SLAM process (5)	42
2.7	Example of pairwise registration of laser scans	42
2.8	Pairwise point cloud registration pipeline	43
3.1	Representation of a Nurbs surface	57
3.2	Least mean squares minimization of the distance between the points and the surface	58
3.3	Nurbs normalization	59
3.4	End-to-end network scheme of our proposal	61
3.5	Confusion matrix for ModelNet10	65
3.6	Training and test precision for ModelNet10	65
3.7	Pairs of confused objects in ModelNet10	66
3.8	Confusion matrix for ModelNet40	68
3.9	Training and test precision for ModelNet40	69
3.10	Pairs of confused objects for ModelNet40	69
3.11	The process to generate a VFD from an input point cloud.	71
3.12	Sample of the division of an instance of the piano class from ModelNet40	73
3.13	Effects of the evolution of Voxel Grid filter in box counting.	73
3.14	Example of least-squares approximation for fractal dimension.	74
3.15	Plot of the calculation of fractal dimension over two vox- elized figures	74
3.16	ModelNet40 samples of calculating fractal descriptor over voxels	75
3.17	Confusion matrix with the one-dimensional fractal descrip- tor for the whole object with a KNN classifier, $k = 5$	77
3.18	Comparison of ModelNet10 results for KNN with different k values and voxel subdivisions using euclidean distance.	78

3.19	Confusion matrix for ModelNet10 using KNN with $k = 1$ and 5^3 subdivisions	78
3.20	Confusion matrix for ModelNet10 with a fully-connected network and 9^3 voxel resolution.	79
3.21	Accuracy before applying Dropout	80
3.22	Accuracy after applying Dropout	80
3.23	Loss before applying Dropout	81
3.24	Loss after applying Dropout	81
3.25	Confusion Matrix for ModelNet10 with SVM and 5^3 subdivisions	82
3.26	Accuracy for SVM classifier in ModelNet40 with different voxel resolutions.	82
3.27	Confusion matrix for ModelNet40 with 7^3 subdivisions and SVM classifier	85
3.28	Errors with least-mean square minimization for box counting fractal dimension calculation.	86
4.1	Overall pipeline of the proposal	89
4.2	Examples of registration	92
4.3	Sample image for classifier testing	93
4.4	Comparative plot between classification alternatives	94
4.5	RGB images of the object-on-robot recognition experiment	100
4.6	Incremental registration of the scene of the object-on-robot recognition experiment	101
4.7	Objects recognised in the object-on-robot recognition experiment	101
4.8	Probability log for the microwave detection	102
4.9	RGB images of the scene of the experiment on the recognition of multiple instances of objects of the same class	103
4.10	Incremental registration of the scene of the experiment on the recognition of multiple instances of objects of the same class	103
4.11	Objects recognised in the experiment on recognition of multiple instances of objects of the same class	104

4.12	Probability log for the experiment on the recognition of multiple instances of objects of the same class	104
4.13	Description of the OGT and SOD algorithms pipelines . . .	107
4.14	Diagram describing the method to compute the transformations between the AAL and the Robot	111
4.15	Common pattern to calibrate both systems in the same coordinate frame	112
4.16	Results of detection of the potentially dangerous area in an office environment	114
4.17	Results of detection of the potentially dangerous area in an office environment	117
4.18	Results of detection of potentially dangerous area on a home environment	118
4.19	Architecture of the proposal	121
4.20	Global scheme of the Navigation Stack	126
4.21	The actual plan of a test house.	128
4.22	Numbered nodes of the house's graph representation.	128
4.23	Map generated for the MPS with the correspondent graph superimposed	133
4.24	Adaptive Monte Carlo Localization	133
4.25	Estimated planes	137
4.26	Shadow effect and specular surfaces issues	137
4.27	Iro Laina's approach	138
4.28	RMSE of the distance to the real plane of the points inferred by Iro Laina.	139
4.29	FusionV1 artifacts	140
4.30	RMSE of the distance to the real plane of the points fused by FusionV1.	141
4.31	Scheme of the new fusion method FusionV2.	142
4.32	RGB images for plane comparisons	143
4.33	Plane comparisons for this experiment	144
4.34	RMSE distances of the point clouds to the fitted plane . . .	145
4.35	RMSE distances to the real plane for this experiment. . . .	146

5.1	Pharos robot in a pilot study at the residence for the elderly Doña Rosa (Alicante)	154
5.2	Hobbit robot in a pilot study at the Doña Rosa senior care home.	154
5.3	RAMCIP robot in a pilot study at a user's home	155
5.4	Rudy in a pilot study	156
5.5	Stevie II in a pilot study	156
5.6	Interaction between a child with autism and a therapist . .	157
5.7	Experiment measuring the interaction of patients with autism with robots and humans	162
5.8	Experiments of interaction with patients with autism using smile measurement as feedback	163
5.9	Conversational robot implemented with Nao	164
5.10	Flowchart of EVA system. Note that the cloud-based parts are shown in a cloud shape.	168
5.11	Trainer's mat	169
5.12	2D poses	171
5.13	3D human pose estimation system	172
5.14	Person detector YOLOv3	173
5.15	Performance of the Human Mesh Recovery network	173
5.16	Joint positions considered by the system	174
5.17	Representation of joints in body coordinates for two different bodies.	177
5.18	Screenshot of the main menu of the developed application .	178
5.19	A sample of a patient performing a rehabilitation session . .	179
5.20	Replay being displayed	180
5.21	Estimated human 3D poses	182
5.22	Accumulated percentage of samples per error for 2D points	183
5.23	Accumulated percentage of samples per error for 3D points	183
5.24	Some sample activities	185
5.25	Diagram of the proposal	189
5.26	Different moments of a recreation for the proposed challenge	193
5.27	Accuracy and F1-score results for OUR's testing videos using ResNet50 with frame skip from 0 to 20.	194

5.28	Accuracy and F1-score results for OUR's testing videos using VGG16 with frame skip from 0 to 20.	195
5.29	Accuracy and F1-score results for OUR's testing videos using MobileNetV2 with frame skip from 0 to 20.	196
5.30	Accuracy and F1-score results for KARDs testing videos using ResNet50 with frame skip from 0 to 20.	197
5.31	Accuracy and F1-score results for KARD's testing videos using VGG16 with frame skip from 0 to 20.	198
5.32	Accuracy and F1-score results for KARD's testing videos using MobileNetV2 with frame skip from 0 to 20.	199
5.33	Evolution of the mean accuracy results for the classifiers and DL networks	200
5.34	Random results of the proposed approach	200
5.35	Time consumed to generate the models from OUR's and KARD's training videos using frame skip from 0 to 20.	201
5.36	Random results of the proposed approach	201
A.1	Sensor ASUS Xtion Pro Live	216
A.2	Nube de puntos de Xtion	216
A.3	Robot Pepper	218
A.4	Dimensiones del Robot Pepper	218
A.5	Sensores y actuadores de Pepper	219
A.6	Articulaciones del Robot Pepper	220
A.7	Convención de signos para los sistemas de coordenadas del Robot Pepper	220
A.8	Rango de giro de la cabeza del Robot Pepper	220
A.9	Rango de giro de la extremidad izquierda del Robot Pepper	221
A.10	Rango de rotación de la extremidad derecha del robot Pepper	221
A.11	Rango de oscilación de la cadera de Robot Pepper	221
A.12	La mano del robot Pepper	222
A.13	Esquema de la base del Robot Pepper	222
A.14	Capturas de Pepper y una Xtion independiente	224
A.15	Ejemplo de simulación de Gazebo	225
A.16	Ejemplo de Gmapping	226

A.17 Ejemplo de visualización PCL	228
A.18 Comparativa entre los casos de mesita de noche y cómoda en vista frontal y lateral	230
A.19 Comparativa entre instancias de <i>copa</i> y <i>jarrón</i> en vista fron- tal y superior.	231
A.20 Comparativa entre instancias de planta y maceta en vista frontal y superior.	231
A.21 Ejemplos de imágenes de KARD	234



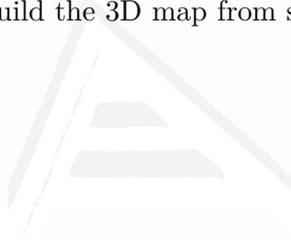
Universitat d'Alacant
Universidad de Alicante

List of Tables

1.1	Activities included in the KARD	21
3.1	Comparative of accuracy results on ModelNet benchmark	83
4.1	Transition matrix between node types.	123
4.2	Mean error per distance to the plane reached by the robot.	136
4.3	RMSE distances to the fitted plane for this experiment	145
4.4	RMSE distances to the real plane for this experiment	146
4.5	Density comparisons for this experiment	146
5.1	Robots used in autism therapies.	159
5.2	Robots used in autism therapies.	160
5.3	Confusion matrix for the DTW Scoring experiments with the KARD.	185
5.4	Results for the known-unknown classifiers and the whole recognition system	199
A.1	Actividades incluidas en el KARD	233

List of Algorithms

1	Pairwise point cloud registration algorithm	91
2	Algorithm for cutting points to be made on the scene	96
3	Algorithm for selection between objects of the same class . .	97
4	Algorithm to build the 3D map from semantic labels	99



Universitat d'Alacant
Universidad de Alicante

Introduction

This first chapter introduces the topics of this thesis. The chapter is organised as follows. Section 1.1 introduces the motivation of this work. Next, in Section 1.2, the different tools used during the development of the thesis are shown. In Section 1.3, the proposal and goals are presented. Finally, Section 1.4 shows the structure of the dissertation.

1.1 Introduction and Motivation

The present thesis discusses three main topics. First, 3D object recognition over point clouds is discussed. Then, mapping of the environment is covered. Finally, other topics related to social robotics are included.

Tridimensional object recognition methods are focused on classifying objects represented in 3 coordinates, just as in the real world. These methods differ from the 2D counterpart in that they focus on topological aspects of the data rather than their visual aspect. Although 2D recognition has a high hit rate and is perfectly applicable in many situations, there are situations where it does not work properly. For example, it has great difficulty in differentiating objects that look very similar in appearance, such as differentiating a photograph from a real object. That is why 3D object recognition can complement and overcome these circumstances. On the one hand, traditional methods that perform this task are very dependent on the conditions of the dataset, as they are largely handicapped by occlusions and noise, and sometimes lack generalisability. On the other

hand, there are several methods that rely on deep learning to perform the classification. Despite their good results, they still suffer in the area of explainability, i.e. how easily their decisions can be understood by a human being. We will explore new methods to carry out this classification that can be more organic and understandable.

Regarding to environment mapping, these systems address the problem of acquiring spatial models of physical environments that can be used by mobile robots. As the aim of this work is to be used by a social robot, we will focus only on indoor mapping techniques. There are good and robust methods that assume that the environment is static, structured and limited in size. However, working with large-scale, unstructured, dynamic maps of the environment is still a problem far from being solved. Historically, research has been divided between metric and topological maps. Metric maps store the geometric properties of the environment, while topological maps describe the connections between different places in the environment. In this thesis, we will try to combine the benefits of these two types of maps by mapping the environment to include semantic information about the objects in the scene. In this way, a robot will be able to navigate the environment and interact properly with those objects it needs to perform its programmed task.

In regard to social robotics, in this thesis we include a comprehensive review of the state of the art of social robots, especially focused on interaction with elderly people and people with autism. According to the World Health Organization, people with disabilities are particularly vulnerable to deficiencies in services, such as health care, rehabilitation, support, and assistance. In this sense, recent technological developments can mitigate these deficiencies, offering less-expensive assistive systems to meet users' needs. On the other hand, in this work, we propose an augmented reality system to help in the adherence and supervision of rehabilitation sessions at home, which can be carried out with low-cost sensors and can therefore be accessible to a large number of patients. This system stores user statistics and allows therapists to adjust exercises according to the results obtained. Finally, in this thesis, we present a visual recognition module that allows robots to identify the people they are interacting with in order

to offer them a personalised treatment. The main idea of this work is that the robot is able to learn new identities in real time with a small interaction with the user, and from that moment to be able to recognise the user unequivocally on other occasions.

It is important to remark that this thesis has been carried out within the framework of the following projects:

- *RETOGAR: A system to enhance the autonomy of the brain injured and handicapped persons for their integration in society.* Founded by the Ministerio de Economía de España and supported by FEDER funds. DPI2016-76515-R [Garcia-Rodriguez et al., 2020].
- *A2HUMPA: Aprendizaje y análisis de comportamientos humanos para monitorización, asistencia personalizada y detección temprana de dolencias.* Funded by the Ministerio de Ciencia e Innovación de España and supported by FEDER PID2019-104818RB-I00.

Also, under the following grants:

- *FPU grant for PhD studies.* Supported by Gobierno de España, Ministerio de Ciencia e Innovación. FPU16/00887.

1.2 Methodology

This chapter includes an explanation of the tools, both hardware and software, used in the project, as well as the frameworks and datasets used to train and validate the results.

1.2.1 ASUS Xtion Pro Live



Figure 1.1: ASUS Xtion Pro Live Sensor. Retrieved from [McGlaun, 2011]

This is a 3D camera aimed primarily at developers. It consists of an infrared sensor, microphones to capture ambient sound, and an RGB camera that allows the depth image to be coloured.

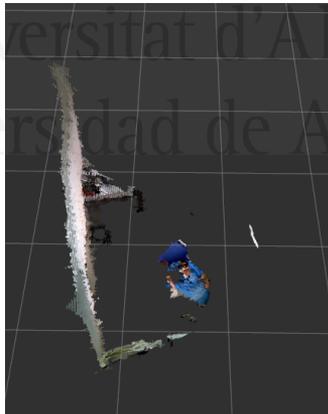


Figure 1.2: View of a point cloud obtained from Xtion registered with the colour information.

Its maximum resolution is 640x480, the depth range it detects is between 0.8 and 3.5 metres and its field of view is 58° horizontally and 45° vertically.

1.2.2 Pepper Robot

Pepper is a humanoid robot created by Aldebaran Robotics and Soft-Bank designed to be a social robot that analyses expressions and voice tones. It was presented at a conference in June 2014 and commercialised in February 2015. As seen in Figure 1.4, the robot has a height of 1.21 metres, a width of 480 mm and a depth of 425 mm.

Its head has 4 microphones (Figure 1.5(a)), two speakers (Figure 1.5(b)), two RGB cameras - in its mouth and forehead (Figure 1.5(c)) - and an Xtion depth sensor behind its eyes (Figure 1.5(d)). It has a gyroscope on its torso and tactile sensors on its head and the back of its hands. Its mobile base consists of 2 sonars (Figure 1.5(e)), 6 lasers (Figure 1.5(f)), 3 shock sensors and a gyroscope. Thanks to these elements, it can adequately perceive the environment around it in order to interpret it and act accordingly.

As for the motor part of the robot, since it is a humanoid robot, it has the same joints as a person, except for the impossibility of moving the fingers independently and the replacement of the lower limbs by a mobile base, as shown in Figure 1.6. In this way, stability with respect to other biped humanoid robots is greatly guaranteed, facilitating their mobility.

Given a joint connecting two parts of the robot, the part of the robot closest to the trunk is considered to be fixed and the part furthest away is considered to rotate around the joint axis. For each of the joints a coordinate system is established so that at robot position 0 they all have the same orientation. Figure 1.7 shows the rotation corresponding to *roll*, *pitch* and *yaw* according to the established convention.

The following figures show the range of rotation of the head (Figure 1.8), the left (Figure 1.9) and right (Figure 1.10) limb joints and the hip (Figure 1.11).

In the hands, the robot has 5 fingers (Figure 1.12) but can only open or close them due to the presence of a single motor per hand.

Finally, at the base (Figure 1.13) it has 3 wheels that allow it to perform translation movements on the X and Y axes and rotation on the Z axis (Yaw) [Aldebaran-Robotics, 2014].

For its programming, Pepper has a visual programming suite, *Choregraphe*, which works by linking boxes. It is a very intuitive environment



Figure 1.3: Robot Pepper from Rovit's research lab

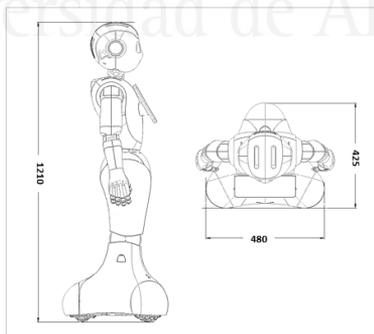


Figure 1.4: Dimensions of the Pepper Robot. Retrieved from [Aldebaran-Robotics, 2014]

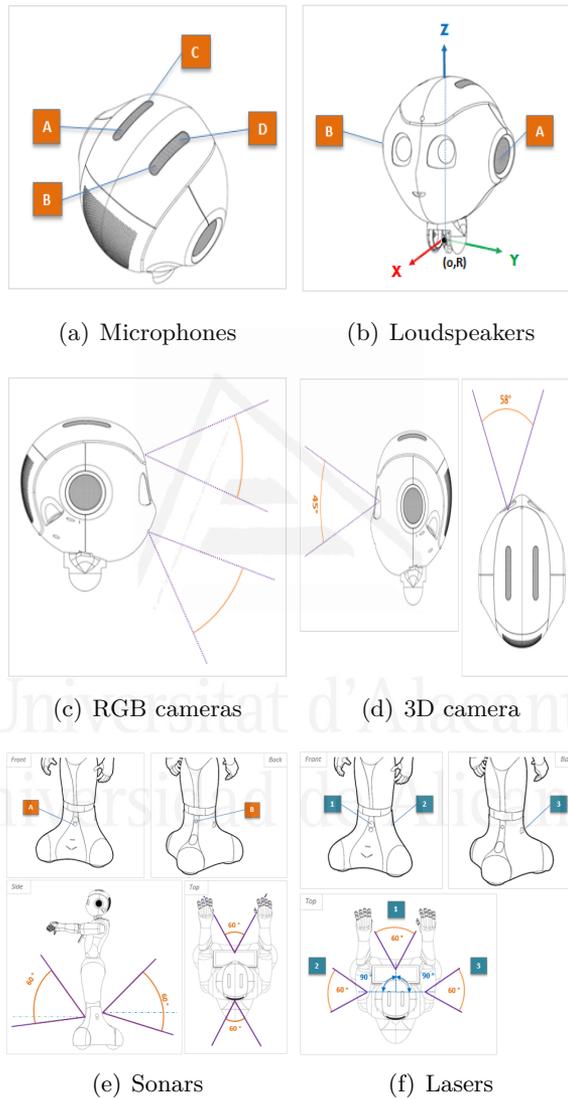


Figure 1.5: Pepper sensors and actuators. Retrieved from [Aldebaran-Robotics, 2014]

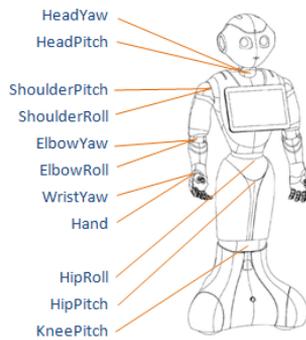


Figure 1.6: Joints of the Robot Pepper. Retrieved from [Aldebaran-Robotics, 2014]

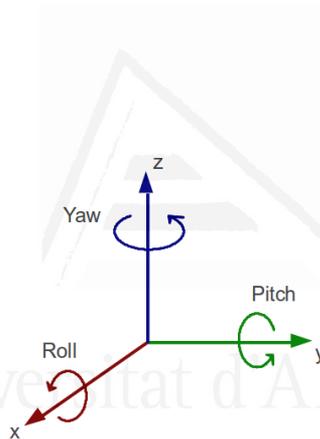


Figure 1.7: Sign convention for Robot Pepper coordinate systems. Retrieved from [Aldebaran-Robotics, 2014]

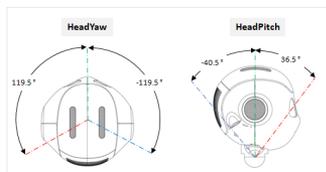


Figure 1.8: Turning range of the Robot Pepper's head. Retrieved from [Aldebaran-Robotics, 2014]

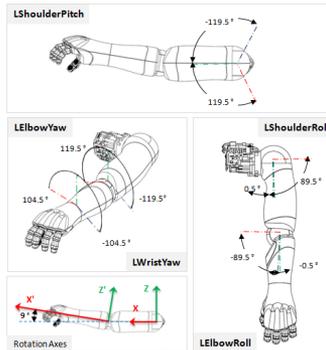


Figure 1.9: Turning range of the left limb of Robot Pepper. Retrieved from [Aldebaran-Robotics, 2014]

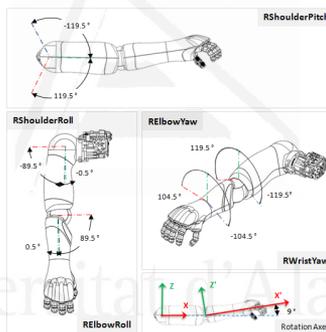


Figure 1.10: Robot Pepper's right limb rotation range. Retrieved from [Aldebaran-Robotics, 2014]

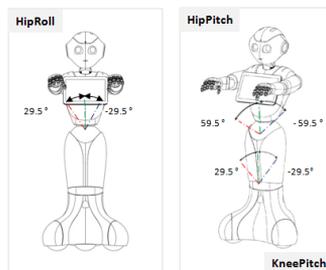


Figure 1.11: Robot Pepper's hip swing range. Retrieved from [Aldebaran-Robotics, 2014]

that allows us to control the robot's movements, play audio or video, display multimedia content via its tablet and respond to visual, auditory or tactile stimuli.

In addition, *Aldebaran Robotics* provides advanced programmers with a development kit, *NAOqi*, available for both Python and C++. It is based on a proxy-based architecture whereby modules communicate with each other by sending messages to services offered by other modules and receiving their response.

A third possibility for programming the Pepper robot is to work with the Robot Operating System (ROS) interface developed by its community. This interface provides a bridge between *NAOqi* and ROS, transforming messages from the former into messages that can be used by the latter [ROS, 2014].

1.2.2.1 The issue with the 3D sensor of the Pepper robot

Most of the experiments were carried out with the robot Pepper and many methods use the 3D information from the depth camera. The robot has three different hardware versions. The latest 1.8 version includes a brand-new three-dimensional sensor. In the newer robot version, the depth sensor was replaced by two 4 MPx color cameras, and a stereo algorithm was used to provide depth perception. Outcome depth maps are generated at 15 fps and have a resolution of 1280×720 . However, versions 1.6 and 1.8a, are both equipped with the defective depth sensor, Asus Xtion. The Asus Xtion depth sensor can provide depth maps with a resolution of 320×240 at 20 frames per second.

As mentioned, the Pepper Robot has an Asus Xtion sensor as a depth camera. Theoretically, this camera can provide accurate depth maps of the scenes. However, the Xtion camera mounted on this Robot appears to provide erroneous depth maps, which also results in incorrect point clouds.

As shown in Figure 1.14, this issue lies in a distorted representation of tridimensional space. The resulting point clouds reveal a wave-like pattern throughout the scene. This issue becomes more evident when it represents plane artifacts, like walls or floors, but it occurs throughout the whole scene. In addition, we noticed that the distortion worsens with increasing

depth, that is, objects near to the sensor were less affected than those further away.

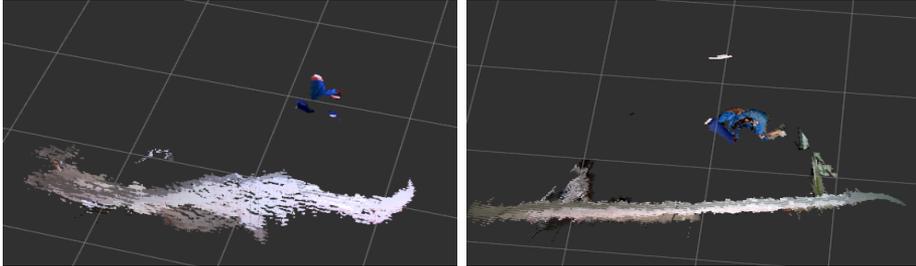


Figure 1.14: These images show an external standalone Xtion point cloud and the Pepper Robot’s Xtion camera. Both images are taken from the same point of view and represent the same scene. The white artifact is a planar surface, which, in this case, is a wall.

It should be noted that this is not an isolated problem or a flaw in our unit. To make sure of them, we have personally contacted different researchers from different laboratories and universities who have reported the same issue with the robot with the same version.

1.2.3 ROS

ROS is a framework for robot software development that functions as an operating system within a heterogeneous cluster.

ROS provides the services of an operating system such as hardware abstraction, low-level device control or message passing between processes. It is designed under a graph architecture in which the processing is distributed and performed on each of the nodes, which can receive and send different types of messages according to their needs.

ROS has been developed under a free Berkeley Software Distribution (BSD) software licence, which allows free commercial and research use, and has a large community of developers who contribute software packages with different utilities for robotics. [Quigley et al., 2009b]

1.2.4 Gazebo

Gazebo is a 3D dynamics simulator designed to accurately reproduce the dynamic environments a robot may encounter. All simulated objects

have mass, velocity, friction, and numerous other attributes that allow them to behave realistically when pushed, pulled, knocked over, or carried. Robots are dynamic structures composed of rigid bodies connected by joints. Forces, both angular and linear, can be applied to surfaces and joints to generate movement and interaction with the environment [Koenig and Howard, 2004].

To achieve ROS integration with stand-alone Gazebo, a set of ROS packages named `gazebo_ros_pkgs` provides wrappers around the stand-alone Gazebo. They provide the necessary interfaces to simulate a robot in Gazebo using ROS messages and services.

Figure 1.15 shows an example of a simulated indoor environment in Gazebo.

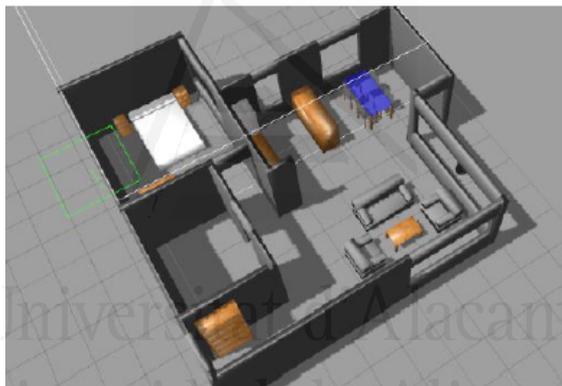


Figure 1.15: Example of Gazebo simulation. Annie’s house. Extracted from [Cazorla et al., 2016]

1.2.5 Gmapping

Gmapping is a laser-based Simultaneous Localization and Mapping (SLAM) algorithm part of OpenSLAM [Grisetti et al., 2007]. This approach uses a particle filter in which each particle carries an individual map of the environment. Authors present adaptive techniques to reduce the number of particles in a Rao-Blackwellized particle filter for learning grid maps, taking into account not only the movement of the robot but also the most recent observation. Figure 1.16 shows an example of its execution.

This algorithm is introduced into ROS in the form of a wrapper in a node called `slam_gmapping`. Using this algorithm, 2-D occupancy grid maps can be created from laser and pose data collected by a mobile robot.

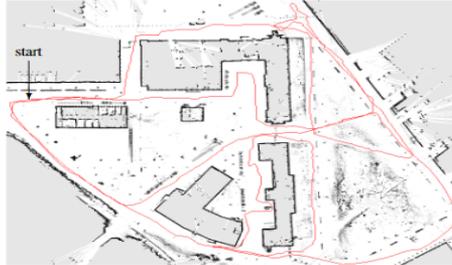


Figure 1.16: Gmapping example of Freiburg Campus. Extracted from [Grisetti et al., 2007]

1.2.6 Caffe

Convolution Architecture For Feature Extraction (Caffe) is a free software framework, under BSD licence, for Deep Learning, initially developed by the Berkeley Artificial Intelligence Research Laboratory (BAIR) and currently supported by a large community of developers. It is developed for C++, Python and Matlab.

It has an expressive architecture that allows for innovation and applicability. The configuration of its models does not require complex programming and allows the use of Graphics Processing Unit (GPU) to accelerate the training and recognition process.

The development of its code has been done in a modular way, which has greatly facilitated the contributions of the community of developers involved in the project, as well as the ease of use for end users.

Caffe also provides predefined and optimised models based on datasets provided by large object recognition challenges such as ImageNet, whose configurations have been successful in these challenges, resulting in high hit rates.

1.2.7 Tensorflow and Keras

TensorFlow is an end-to-end open source platform for machine learning for Python and C++. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in Machine Learning (ML) and developers easily build and deploy ML-powered applications [ten,]. It is one of the most popular Deep Learning (DL) frameworks, as it provides great ease of use and allows the implementation of architectures with different degrees of abstraction, depending on the user's level of expertise.

Keras is an API designed for human beings, with a high level of abstraction. Keras follows best practices for reducing cognitive load: it offers consistent and simple APIs, it minimizes the number of user actions required for common use cases, and it has extensive documentation and developer guides [Chollet et al., 2015]. Currently, this system is fully integrated within TensorFlow.

1.2.8 Darknet

Darknet is an open source neural network framework written in C and Compute Unified Device Architecture (CUDA). It is fast, easy to install, and supports Central Processing Unit (CPU) and GPU computation [Redmon, 2016]. Its use is not as popular as frameworks such as Keras or Tensorflow, but it is still used as some famous deep learning architectures were implemented with it. This is the case of You Only Look Once (YOLO) [Bochkovskiy et al., 2020], which is one of the most widespread state-of-the-art methods for object recognition in images.

1.2.9 PCL

Point Cloud Library (PCL) is a free BSD-licensed library used for 3D point cloud processing. It is a library written entirely in C++ templates with the aim of achieving the highest computational efficiency and performance, with most of the mathematical operations implemented using the Eigen library [Guennebaud et al., 2010], an open source library for linear algebra, and with compatibility with *OpenMP*.

From an algorithmic perspective, this library incorporates a large number of 3D processing algorithms that work with point clouds, including: filtering, feature estimation, surface reconstruction, segmentation or registration, among others. Each set of algorithms is defined by base classes that attempt to integrate all common functionality, allowing for extensibility and compact and clean use of algorithm implementations.

PCL has full compatibility with ROS, so that the algorithms of this library can be deployed as ROS nodelets and operate as if they were another node of the framework.

As for the visualisation of the data, PCL comes with its own visualisation library, based on The Visualization Toolkit (VTK) [Schroeder et al., 2005], which provides support for rendering 3D point clouds and basic 3D [Rusu and Cousins, 2011] shapes.

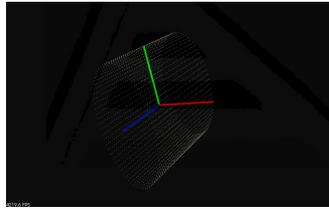


Figure 1.17: PCL display example. Retrieved from [PointClouds, 2015]

1.2.10 Datasets for 3D recognition

There are some state of the art datasets that have captured single objects in isolation with real sensors for 3d object recognition, as presented in [Firman, 2016]. *RGBD Object Dataset* [Lai et al., 2011] and *MV-RED* [Liu et al., 2015] offer captures of 300 and 500 objects using Kinect v1, but without information of the pose. *BigBIRD dataset* [Singh et al., 2014] and *YCB Object and Model Set* [Calli et al., 2015] additionally incorporate pose information of 100 objects using an Asus Xtion Pro sensor. However, the biggest effort has been performed by *A large dataset of object scans* [Choi et al., 2016] that provides captures of more than 10000 objects using a PrimeSense sensor, with the camera pose computed directly from the data. There also are some other widely used datasets such as SUNRGBD

[Song et al., 2015], SCANNET [Dai et al., 2017] and S3DIS [Armeni et al., 2017]. Nonetheless, these datasets are intended for object detection and segmentation, so it is unclear how to adapt them to enable a fair testing protocol for benchmarking object classification approaches.

The main problem with the real data is the difficulty to acquire ground truth for 3d pose estimation, that cannot be obtained without external hardware. To solve this issue, many researchers have adopted synthetic datasets. These datasets allow users to control several aspects more carefully, such as point density, pose or noise level. ModelNet [Wu et al., 2015a] provides two different datasets *ModelNet10* and *ModelNet40* with thousands of CAD models for 10 and 40 different classes respectively. They are considered de facto standard datasets for 3d object recognition tasks. On the other hand, *ObjectNet* [Xiang et al., 2016], with 100 categories and more than 40 thousand aligned 3D shapes, represents another of the biggest datasets for this purpose.

1.2.10.1 Princeton ModelNet

Princeton ModelNet project [Wu et al., 2015b], is one of the most widely used benchmarks for 3D object recognition. This dataset has two versions: ModelNet10 and ModelNet40.

ModelNet10 offers a set of more than 4,700 CAD models from 10 different categories that are manually aligned, and divided into training and test set. Following the steps explained in [Garcia-Garcia et al., 2016], we converted these models into Point Cloud Data (PCD) clouds, compatible with the PCL library. In [Garcia-Garcia et al., 2016] we can also see the the highly imbalanced distribution of both training and test sets.

The main problem of this dataset is the visual similarity between categories, that mainly occurs with the *night_stand* y *dresser* categories. In Fig. 1.18 we can see samples of both categories, which are very difficult, even for a human, to distinguish them. It is clear that there are other designs which are more differentiable from each other, but this problem occurs quite often, thus leading to a potential decrease of the accuracy of the classification results.

ModelNet40 offers a set of more than 11,000 CAD models from 40

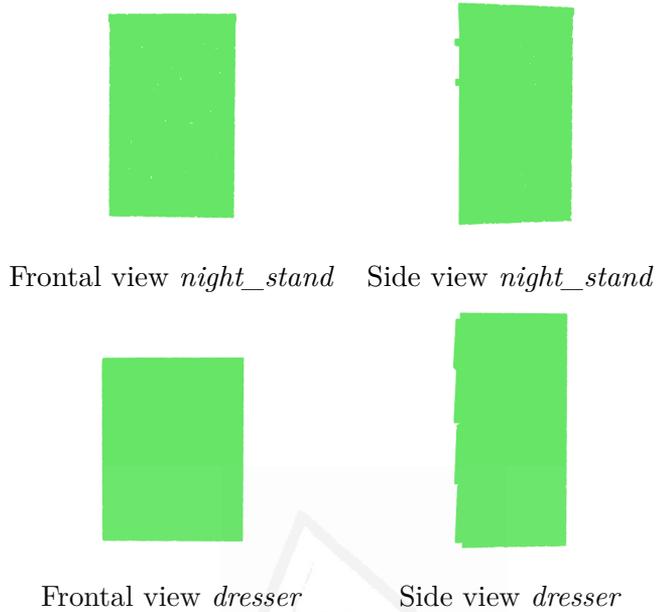


Figure 1.18: Comparative between instances of *night_stand* and *dresser* in frontal and side view

different categories manually aligned, and divided into training and test set. Similarly to the ModelNet10 dataset, we converted these models into PCD clouds. This version of the dataset also shows a high imbalanced distribution of both training and test sets.

The problem referred to ModelNet10 happens with this dataset too, as shown in Figures 1.19 and 1.20, as it is of the same nature. In addition to those previously mentioned, we will see frequent confusions between *door* and *curtain* classes, because they are both plain rectangles, taller than wide and shallow. Something similar occurs between *bench* and *desk*, because most of their instances consists of a large rectangular base with four legs. However, main confusions are between classes *flower_pot*, *plant*, *vase* and *cup*. We can find many potted plants in *flower_pot* and *plant* instances indistinctly, so the classification is expected to be somewhat arbitrary in this case. On the other hand, *flower_pot*, *vase* and *cup* categories have very similar instances that only differ in scale, color and use, feature that are not commonly taken into consideration in 3D object recognition algorithms.

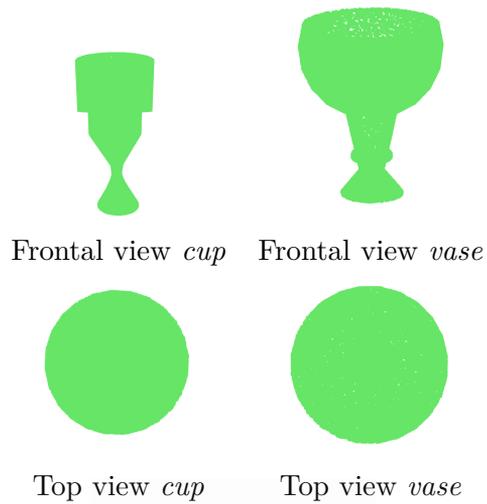


Figure 1.19: Comparative between instances of *cup* and *vase* in frontal and top view.

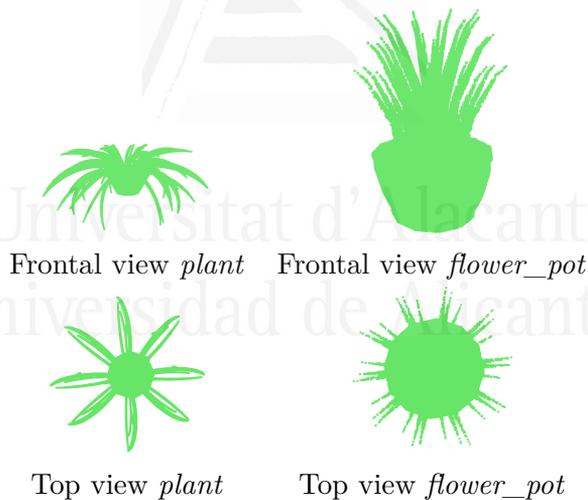


Figure 1.20: Comparative between instances of *plant* and *flower_pot* in frontal and top view.

1.2.11 Datasets for action recognition

Although this thesis does not perform recognition of actions in use, an analysis of the suitability of physical rehabilitation exercises has been carried out by comparing the similarity between the sequence of model

poses, performed by a trainer or therapist, and that performed by patients.

For this, we need labelled action and pose data to be able to evaluate the effectiveness of the comparison metrics proposed in our work. In the state of the art there are different datasets for the realisation of this task.

- Moments in Time [Monfort et al., 2019] is a large-scale dataset for the recognition of actions and events in videos. It includes a collection of one million 3-second labelled videos of people, animals, objects and natural phenomena.
- AVA dataset [Gu et al., 2018] includes the annotation of 80 atomic actions, instead of composite actions, distributed over 350,000 videos, annotated with their corresponding location and date of recording, for a total of 1,600,000 tagged actions.
- Kinetics [Kay et al., 2017] is a large-scale, high-quality dataset of YouTube video URLs that include a wide range of human-centric actions. It consists of approximately 300,000 video clips and covers 400 classes of human actions with at least 400 video clips for each action class. Each clip is about 10 seconds long and is labelled with a single class. All clips have undergone multiple rounds of human annotation and each clip comes from a single YouTube video. The actions span a wide range of classes including human-object interactions, such as playing instruments, as well as human-human interactions, such as shaking hands and hugging.

Due to the specific needs of our work, we will use the dataset Kinect Action Recognition Dataset (KARD) [Gaglio et al., 2015], as it provides us with the skeletons of the persons and allows us to directly apply our similarity assessment metrics between poses.

1.2.11.1 KARD

KARD [Gaglio et al., 2015] contains 18 different short-time activities, summarized in Table 1.1, which overall match the features of some rehabilitation exercises. It involved 10 individuals repeating 3 times each activity.

The dataset provides the video of each person performing an activity and the position of the body joints in both 2D and 3D coordinates as captured by a Kinect device. In addition, it provides the corresponding depth maps. Examples of images from this dataset are shown in Figure 1.21

Table 1.1: Activities included in the KARD [Gaglio et al., 2015].

ID	Activity	ID	Activity	ID	Activity
1	Horizontal arm wave	7	Draw tick	13	Hand clap
2	High arm wave	8	Toss paper	14	Walk
3	Two hand wave	9	Forward kick	15	Phone call
4	Catch cap	10	Side kick	16	Drink
5	High throw	11	Take umbrella	17	Sit down
6	Draw X	12	Bend	18	Stand up



Figure 1.21: Image examples from KARD. Extracted from [Gaglio et al., 2015]

1.3 Proposal and goals

The common nexus of this thesis is the research in the field of social robotics. To this end, great efforts have been devoted to the tasks of 3D object recognition and environment mapping, as these are two fields of vital importance for the interaction of the robot with a domestic environment. Additional research has also been carried out to provide the robot with the ability to recognise people, and a rehabilitation support system has been developed that can be incorporated into the robot and that can help with adherence to treatment.

Specifically, the goals of this research work are the following:

- Explore different alternatives for 3D object recognition.
- Finding new ways of mapping the environment including useful information about objects in the environment.
- Enable the robot to perform personalised treatment with the people it interacts with and to assist them in rehabilitation tasks.

1.4 Structure of the dissertation

In accordance with the topics presented in the introduction, the structure of the thesis is as follows.

In Chapter 2, **Background**, the main state-of-the-art works on 3D object recognition and environment mapping are collected. Also, important information related to social robotics is included, which will be extended later in the thesis.

In Chapter 3, **3D Object recognition**, different approaches to 3D object recognition are explored, which are novel alternatives to current state-of-the-art methods. Free surfaces and mathematical concepts such as fractals will be used to obtain 3D descriptors.

In Chapter 4, **Mapping of the environment**, different solutions are explored to provide a social robot with knowledge about the surrounding environment. These approaches include a mapping of the environment that includes information about the objects present in the scene, a route planner that allows the optimisation of routes between different rooms of the house, and a mapping that allows the assessment and detection of possible risks to which a person may be exposed inside a house. This chapter also proposes a solution to the Pepper robot's depth sensor problem, based on sensory fusion and depth inference from colour images.

In Chapter 5, **Investigations applied to social robotics**, contributions related to social robotics that do not fit directly into the previous sections are included. Firstly, we provide an extensive review of social robotics applied to the field of care for the elderly and for therapy with people with autism. Secondly, we present an augmented reality system that allows physical rehabilitation therapies to be carried out at home, and which provides professionals with feedback that allows them to evaluate the results. Finally, we present a human recognition module capable of learning identities in real time, which allows personalising the robot's interaction with humans.

In Chapter 6, **Conclusions**, the final conclusions and considerations are drawn. Moreover, contributions to the topic are presented and publications derived from this work are enumerated. To conclude the chapter,

future directions of the research carried out are shown as well.



Universitat d'Alacant
Universidad de Alicante

Background

In this chapter, we describe the most relevant related works about object recognition, mapping of the environment, and social robotics for personal assistance. A brief introduction is provided in Section 2.1. Then, in Sections 2.2 and 2.3, a review of the most relevant state-of-the-art methods for 2D and 3D object recognition, respectively, is given. Section 2.4 then presents a selection of works that have used the mathematical concept of fractals to perform classification tasks. A review of methods for mapping an indoor environment follows in Section 2.5. Finally, Sections 2.6 and 2.7 provides a brief review of social robotics, focusing mainly on personal assistance tasks and rehabilitation, and Section 2.8 reviews the state of the art on re-identification of people, an important aspect of social robotics to enable personalised treatment in their interactions with humans.

2.1 Introduction

A brief review of the state of the art allows us to put the work carried out in this thesis into context. To this end, we not only explain the characteristics of the previously published works, but also seek to analyse them and detect their main limitations that can be improved with our contributions.

To this end, the following sections review the fields of object recognition and environment mapping, the main topics of the work, as well as a brief

introduction to social robotics, especially focused on personal assistance tasks.

2.2 2D Object recognition

Object recognition is the field of computer vision dedicated to the search for an identification of objects in an image. Humans have an innate ability to recognise objects with little effort despite varying aspects such as point of view, scale, translation or rotation. However, this task is still a huge challenge for artificial intelligence and has been the subject of much research for decades.

Object detection refers to the identification of the location where a certain component of a general object class is located. For example, looking for faces in an image, finding the eyes or mouth in an image, etc. The term recognition, on the other hand, refers to the classification of an object or subclass within a general class of objects present in a region of the image that has been previously isolated. For example, after detecting a face, recognising the identity of the person it belongs to.

Detection can be seen as a binary classification problem. For each instance, it must be decided whether the object is present or not (object or non-object). On the other hand, recognition becomes a multi-class problem, where it is of interest to identify the specific class to which the identified object belongs.

One possibility to perform recognition would be to train models for each of the object types, run each of them against the image and choose the best fit according to a certain criterion. However, these models would have problems distinguishing similar objects and would be highly inefficient. Therefore, the alternative is to directly train a classifier that includes examples for all classes, which improves the discrimination between the different classes [Amit, 2002].

We will look at the evolution of these techniques in the following subsections, starting with the traditional techniques used in the early days of artificial intelligence (2.2.1 and 2.2.2) and ending with the new approaches resulting from the boom in Deep Learning in recent years (2.2.3).

2.2.1 Model-based

Model-based methods consider that knowledge of an object's appearance comes from explicit modelling of its form, regardless of the context in which the object was found.

Recognition is achieved by finding correspondences between certain features of the image and comparable features of the chosen model. The big question to be solved by such methods is what can be considered as a feature and how it can be used to compare the image and the model. Some of them use global features such as the area or the length of its perimeter. These methods are mostly used in applications where lighting is controlled and no occlusions occur, assuming perfect segmentation between objects and background, which is not applicable in most real cases.

More elaborate methods make use of local features such as corners or junctions. They are distinguished and compared with each other on the basis of criteria such as scope, robustness to noise and occlusions, computational efficiency and correctness. They are therefore the most interesting ones to be analysed.

It is considered that a good form of model representation should be constituted by a combination of local primitives describing a limited portion of the model. Its efficiency and stability are guaranteed as this type of description is only partially affected by occlusions in the image.

To achieve efficient matching in shape descriptions, there must be a limited number of categories denoted by each primitive so that the continuous range of possible shapes is discretised in its representation, such as considering a surface as planar, cylindrical, elliptical or hyperbolic.

A vitally important issue is the choice of good primitives for describing shapes. These primitives must capture the differences that distinguish between different objects and reflect the regularities and structures of the real world. These criteria lead to the conclusion that primitives are highly dependent on the field of application in which you want to work.

Another interesting aspect of these methods is the choice of the coordinate system on which to describe the location of the primitives. There are two main approaches to this task: object-oriented representation and observer-centred representation.

In the case of object-oriented representation, the object is represented as a set of hierarchical parts, in which each part has its own coordinate system and is located from the coordinate system of its parent. Despite its difficulties with partial occlusions, works such as [Dickinson et al., 1992] have succeeded in recovering 3D volumetric primitives from 2D partial views and recognising objects from them.

The alternative is the observer-centred or multi-view representation, which describes the 3D object using a set of characteristic 2D views, each showing the object from a different angle. The pairing is simpler than in the previous case as there is no need to project the model during the pairing and the continuous space of views has been reduced to a discrete space of characteristic views. This method is considered to be the most organic, as studies have shown that the human visual system recognises objects more quickly and accurately when viewed from particular viewpoints [Edelman and Bülthoff, 1992].

To recognise an object in an image, most of these methods perform the following sequence of steps:

- **Primitive detection.** Locate primitives in an image and represent them as symbols.
- **Perceptual organisation.** Identifying stable groupings of primitives.
- **Indexation.** Use the primitives to select the most similar models from the database of known models.
- **Matching.** Find the best match between the image primitives and those of the selected models.
- **Verification.** Decide whether the pairing suggests that the selected model is present in the image.

To recognise any number of objects present in an image, each time an object is recognised its primitives are removed from the image and sent to the next recognition cycle with the remaining primitives. Therefore, the whole process is a large search through each of the stages, which makes it a very computationally expensive process [Pope, 1994].

2.2.2 Appearance-based

In appearance-based methods only the appearance is used, which is usually captured by different two-dimensional views of the object of interest. Based on the features used, we can divide models into two broad classes: local approach or global approach.

A local feature is a property of an object located in a single point or small region of the image. It is a small piece of information that describes a simple but distinctive property of the object's projection in the camera image. Examples of local features of an object can be the colour, gradient or grey value of a pixel or a region. To be used in recognition tasks, local features must be invariant to changes in illumination, noise, changes in scale or viewpoint, so different features are often combined to form a more complex and robust description.

In contrast, global features attempt to cover the information contained in the entire image or patch. These techniques range from simple statistical measures, such as a histogram of features, to dimensionality reduction techniques such as Principal Component Analysis (PCA).

While global methods allow reconstructing the original image and are more robust to enlargements, local methods allow working with partially occluded objects.

The first step for local feature-based systems is to detect regions of interest in an image so that they can be compared to regions of interest extracted from known objects. When the region of interest consists of a single point, this is known as a point of interest or keypoint. These region detectors can be classified into three groups:

- **Corner-based.** They look for points of interest that contain transitional areas in the image, such as corners. However, they do not work on uniform regions or regions with smooth transitions. Their operation is based on the calculation of first and second derivatives in the image as a way to obtain the gradient at each point and thus detect areas of higher contrast. One of the best known is the Harris corner detector [Harris and Stephens, 1988].
- **Region-based.** They search for regions of the image with uniform

brightness. One of the most commonly used methods is MSER (Maximally Stable Extremal Regions) [Matas et al., 2004].

- **Other approaches.** They take into account the entropy of a region or try to mimic human visual attention (saliency). An example of this approach is the ESBR (Entropy Based Salient Region) method [Cheng et al., 2015].

Once the region of interest has been detected, it needs to be described by feature descriptors that must be robust to image variations such as distortions, scale changes, illumination changes or noise.

One of the simplest descriptors is a vector with the pixel intensities of the region of interest, which by calculating cross-correlation can be used for similarity calculation between regions. The main problem is the huge dimensionality of the descriptors for the matching and recognition tasks, which leads to a huge computational burden [Roth and Winter, 2008].

Descriptors can be divided into the following categories:

- **Distribution-based.** Represent the properties of a region using histograms. They typically use geometric properties such as the location and distance of points of interest in the region (corners and edges) and local orientation information (gradients). One of its most popular descriptors is SIFT, which uses the Gaussian difference as a region detector [Lowe, 1999].
- **Filter-based.** They apply a combination of local differential operators with the aim of obtaining rotation invariance mostly. One of the best known methods is that of differential invariants [Olver, 1995].
- **Other methods.** Within these methods we can find cross-correlation, a statistical measure that uses as a descriptor the intensities of the pixels belonging to a region of interest [Lewis, 1995].

Finally, with the descriptors extracted with the previous techniques, a final classification stage is necessary to allow the final recognition of the object.

The most basic classifier available is the K-nearest neighbour classifier. It is a very simple method whose operation is based on assigning an object to the class of most of its nearest neighbours, according to a distance measure that will depend on the type of descriptor used, the most common being the Euclidean distance between vectors.

It can be very interesting to assign weights to neighbours' contributions, so that closer neighbours contribute more than more distant neighbours, according to a distance-weighted function.

The best choice of k depends on the data being treated. Generally, larger values of k are more tolerant to noise, but make class separations more fuzzy. The accuracy of this algorithm can be severely degraded by the presence of noise, irrelevant features or having the features at different scales [Kataria and Singh, 2013].

Another more complex classifier that works very well in practice is the Support Vector Machines (SVM).

To decide the class to which a new point belongs, from data belonging to known classes, each of the points is seen as a p -dimensional vector of which one wants to know how they can be separated by a $(p-1)$ -dimensional hyperplane that assumes the decision boundary between the classes.

There is an enormous number of hyperplanes capable of classifying data. In the case of the Support Vector Machines, the goal is to select the hyperplane that represents the largest separation between the two classes, so the hyperplane whose distance to the nearest point in each class is maximal is selected.

Formally, a Support Vector Machine constructs a hyperplane or set of hyperplanes in a very large, even infinite, dimensional space. The best separation is achieved by the hyperplane that maximises the distance between classes, so the larger the margin the smaller the generalisation error of the classifier [Suykens and Vandewalle, 1999].

2.2.3 Deep Learning-based

One of the fundamental goals of researchers has been to replace the hand-created feature descriptors explained in previous sections, which require expertise in the application domain, with multilayer networks capa-

ble of learning them automatically by means of a general-purpose training algorithm.

The solution to this problem was proposed in the 1970s and opened up a new branch of machine learning: Deep Learning [Werbos and Werbos, 1994].

Deep Learning architectures consist of a hierarchical stack of layers that apply non-linear functions on the input data, using an internal set of weights. The training method makes use of the backpropagation technique to propagate the error gradient from the last to the first layer, so that they can correct their internal weights iteratively.

These networks learn to map an input of fixed size, such as a matrix or a vector representing an image, to another output of fixed size, which in the case of object recognition is usually a vector containing the probability for each possible category [LeCun et al., 2015].

Despite their great potential, these networks fell into oblivion until recent years. Extracting features with little prior knowledge was considered practically impossible. Also, it was felt that the training method based on gradient descent would inevitably get trapped in local minima, which would offer lousy solutions to the problem.

In recent years, interest in Deep Learning has been rekindled by new discoveries. It has been shown that local minima are not a real problem for these deep neural networks, with practical demonstrations in handwritten digit recognition work, greatly improving the results of traditional techniques [Cireřan et al., 2010]. Furthermore, thanks to advances in parallel computing, graphics processing units (GPUs) have made it possible to greatly accelerate the training of these huge networks, which had previously been computationally unfeasible.

These new advances led to the discovery of a new type of neural network that is very important in image processing: Convolutional Neural Network (CNN). The architecture of this network is composed of multiple phases of convolution and pooling layers, using the non-linear filters *ReLU* (Rectified Linear Unit) as an activation function, ending in a phase of convolutional and fully-connected layers that generate the final result. The idea behind layer stacking is the understanding that many natural signals

are hierarchically composed, so that high-level features are obtained by compositing lower-level ones. [Krizhevsky et al., 2012].

In essence, the process of selecting the most relevant features for each problem has been automated, allowing these deep networks to handle the entire recognition process.

2.3 3D Object recognition

Object recognition is a very important topic in the research community, as it can provide the understanding of the scene for an artificial intelligence. At the beginning of computer science, the main focus was in object recognition with images. Deep learning approaches achieved high accuracy levels in many competitions [Deng et al., 2009], even better than humans, with *Convolutional Neural Network* architectures [Krizhevsky et al., 2012].

However, 2D object recognition relies on appearance and not the shape so, for example, it cannot differentiate between a picture and the object that has been depicted inside. With the advent of depth sensors, we can have information about the shape of the object, so we can differentiate between these classes. Nevertheless, the inherited unorganized representation of the 3D information, where the neighborhood of every point cannot be directly extracted, makes this task harder than the 2D counterpart [Luo et al., 2020].

Traditionally, 3D object recognition has been carried out with local and global feature-based descriptors and geometric primitives [Alhamzi et al., 2015, Xia et al., 2020]

On one side, local methods describe a single point and are only based on local geometric features around the points of interest. However, they require methods to select which points are in fact keypoints, and specialized classifiers to deal with multiple feature vectors per object [Bayramoglu and Alatan, 2010].

On the other side, global methods describe the whole object with a single vector, which is directly suitable for standard, traditional classifiers. Nonetheless, they need a segmentation stage in order to isolate the object [Aldoma et al., 2012a]. We consider that global techniques are more in-

teresting than local approaches to be used as benchmark for further novel methods, as they can be directly used with the classifier of choice, so a revision of the most notorious alternatives of this family is provided in the following paragraphs.

Ensemble of Shape Functions (ESF) [Wohlking and Vincze, 2011] is the only one descriptor that do not rely on normals to describe the shape. It is a combination of ten histograms generated from three different shape functions: point distance (D2, line between point-pairs randomly sampled), angle (A3, angle formed by 3 randomly sampled points), and area (D3, area created by three points) [Osada et al., 2001]. The input cloud representing the object is voxelized to approximate the surface and is used to trace the line joining pair of points efficiently. For every function, there are 3 histograms depending on the connectivity of the points that lies on the surface, off the surface or mixed, according to [Ip et al., 2002]. Finally, the distance ratio histogram is create using the lines from D2 function.

Viewpoint Feature Histogram (VFH) [Rusu et al., 2010] is a global descriptor based on the *Fast Point Feature Histogram (FPFH)* local descriptor [Rusu et al., 2009]. It is composed of two elements: a viewpoint direction component and an extended FPFH component. For the viewpoint direction, the centroid of the object is calculated, and the vector between the viewpoint of the camera and the centroid is then computed. Then, for every point in the object the angle between this vector and its normal is computed. The extended FPFH is computed as the original method, using the centroid as the keypoint and the rest of the points as its neighbors.

Clustered Viewpoint Feature Histogram (CVFH) [Aldoma et al., 2011] is an extension of the *VFH* descriptor that calculates a histogram for each region of points, and not for the whole object. These regions are split by removing points with a high curvature value, and then applying a region growing segmentation algorithm.

Finally, *Oriented, Unique and Repeatable Clustered Viewpoint Feature Histogram (OUR-CVFH)* [Aldoma et al., 2012b] is an iteration of the *CVFH* method that defines a robust unique and repeatable reference frame to describe the object that allows avoiding the ambiguity over the camera roll angle.

With the explosion of new Artificial Intelligence (AI) hardware accelerators, many researchers are focusing on deep learning methods for tackling the 3D object recognition task. We introduce these methods by grouping them in three big categories according to their 3D data representations.

Point Cloud. 3D data is represented as a raw unordered point cloud. These methods usually extract features by analyzing the neighborhood of every point within a radius. The most representative proposal in this case is *PointNet++* [Qi et al., 2017]. It generates a feature vector for the whole cloud by applying order-invariant transformations to every point, generating local hierarchical features that are sampled and grouped, and uses it to segment and classify the scene.

Some proposals are based on the previous architecture. This is the case of *VoteNet* [Qi et al., 2019], a novel technique based on Hough voting, that uses *PointNet++* layers as the backbone. This approach selects a set of interesting points, with their corresponding features, as seed points to generate clusters of object instances based on their votes. Finally, these clusters are transformed into 3D bounding boxes with their corresponding categories.

Another work, *SplatNet* [Su et al., 2018], extends the concept of 2D SPLAT images into 3D. It uses hash tables as a efficient implementation of neighborhood filtering, which provides an easy mapping of 2D points into 3D space. Then, bilateral convolutions are used to extract a set of features.

In the case of *SO-Net* [Li et al., 2018a], the authors propose a method to guarantee invariance to point permutations. It builds a Self-Organizing Map (SOM) through modelling the spatial distribution of the point cloud and using the neighborhood of every point to extract hierarchical features. As a final step, this method generates a global feature vector for the whole cloud.

Other approaches, like *Point-Voxel CNN (PVCNN)* [Liu et al., 2019b], combine the sparse representation of the data with voxelized convolutions that increase the performance of the data access and improve the locality of the method. In this work, a new efficient primitive is introduced, Point-Voxel Convolution (PVConv), that converts points into voxel grids,

aggregates neighboring points with voxel-based convolutions and transforms them back to points. In order to obtain features with a higher level of detail, they include point-based feature transformations.

2D projections. In this category of methods, input data is represented as multiple 2D projections of the tridimensional data. Traditionally, they have been the most common approaches, and usually have a 2D *Convolutional Neural Network* to carry out the processing, as presented in [Su et al., 2015]. Some studies train a boosting classifier to group different views and improve the quality of the inference, like [Johns et al., 2016]. On the other hand, another works make a manual choice of the best views to perform the classification. This is the case of [Gomez-Donoso et al., 2017a], that considers 3 orthogonal views to be good enough to carry out the classification task, and feed 3 independent neural networks, whose results are finally combined. Based on ModelNet benchmark, [Wu et al., 2019], these methods have the highest classification performances, but they need multiple views of the object, so they would perform poorly in real-world applications if they have to deal with with occlusions and partial views.

Voxelization. In this case, input data is a discretization of the original point cloud, grouping points into different clusters according to a neighborhood criteria, that serve as a approximation of the original shape. Commonly, every voxel is represented as a binary value, 0 or 1, that indicates the presence of points in the space the voxel represents. *3D ShapeNets* [Wu et al., 2015a], the study carried out by the authors of ModelNet, discretizes the data as a cubic voxel and applies 3D convolutions to generate the features. In a similar manner, *VoxNet* [Maturana and Scherer, 2015] applies a 3D *Convolutional Neural Network* to the volumetric representation to generate the classification. Another works that are based on 3D convolutions are [Garcia-Garcia et al., 2016, Gomez-Donoso et al., 2020b], which also use the voxel representation of the objects. On another note, *PointGrid* [Le and Duan, 2018] creates grid cells with a constant number of points with a point quantization technique, saving points' coordinates to improve the representation of the local geometry of the object. Other works, such as *O-CNN* [Wang et al., 2017] and *OGN* [Tatarchenko et al.,

2017] combine the use of octree representations with the performance of 3D convolutions to lower memory consumption and improve the performance.

Similar to approaching the problem by using multiview 2D projections, works like *MO-VCNN* [Qi et al., 2016] generate different rotations of the 3D models and extract high level features from every pose that are combined into a final feature vector. Despite the popularity of 3D convolutions, other deep learning architectures have been used. *Vconv-dae* [Sharma et al., 2016] uses a convolutional denoising auto-encoder as a feature learning network, [Brock et al., 2016] a *Variational auto-encoder* and [Wu et al., 2016] a *Generative Adversarial Network*.

The main problem of this category of techniques is the loss of detail due to the discretization step, and that require high amounts of memory for the 3D convolutions, which are mainly useless due to the internal void of the representation, namely, that the point cloud only has points in the surface of the object.

2.4 Fractals used for classification purposes

The fractal dimension has been tested as descriptor in the past, with promising results. Regarding the field of plant recognition, in [Bruno et al., 2008] authors researched a range of algorithms for calculating the fractal dimension. They segment the veins of the plants and their borders, and they calculate the fractal dimension using this information, and classify them into different species of *Passiflora*, obtaining an average accuracy of 97,77%. Another example of the use of the fractal dimension as descriptor is [Shen, 2002]. In this work, which is more theoretical, authors propose the box counting algorithm to calculate the fractal dimension. They aim to demonstrate how fractal geometry can describe properties of the urban development that cannot be capture with euclidean geometry. In this paper, they use 2D binary images of maps of 20 different American cities to carry out the analysis. In this study, they conclude that there is a correlation between fractal dimension and the level of urban development of each city, but they found no evidence of a correlation between this dimension and the population density.

As a last example, we can find the article [Gómez et al., 2009], which uses fractal dimension to analyse signals from a magnetoencephalogram. In this case, fractal dimension is considered as an indicator of the complexity of a signal, that is usually calculated following the Higuchi algorithm [Higuchi, 1988]. Authors seek to verify the reliability of the fractal dimension as an Alzheimer detector. Using this technique, they achieve a 87,8% of accuracy and a 95,24% of specificity, with 5 channels, so they obtained good results for diagnosing this disease.

There is a specific feature that appear in each of the methods described above, and that is the fact that all of them work with the fractal dimension in two dimensions. Because of this, in this work we propose a method to carry out 3D object recognition, taking advantage of the properties of the fractals, in the hope of popularizing its use.

2.5 3D Mapping

Building an appropriate representation of the environment in which an autonomous robot operates is still a widely addressed problem in the robotics research community. This problem is usually known as map building or mapping since maps are considered the most common and appropriate environment representation [Thrun et al., 2002]. A map is useful for robot localization, navigation [Booi et al., 2007] and path-planning tasks [Bhattacharya and Gavrilova, 2008], but also for a better understanding of the robot's surroundings [Pronobis et al., 2009]. That is, a map may not be limited to metric (e.g. specific poses of objects/obstacles) and topological information (e.g. paths from one place to others), but it can also integrate semantic information (e.g. symbolic representations of objects, expected behaviours for specific locations, or even situated dialogues, to name a few) corresponding to the objects, agents, and places represented on it.

2.5.1 Visual-inertial SLAM

SLAM is one of the fundamental challenges of mobile robotics, which focuses on the need to build a map of the environment while determining

the robot's location within it.

Initially, the robot's map and position are not known, the robot has a known kinematic model and moves through an unknown environment with known artificial or natural landmarks. To perform the task, the robot must be equipped with a sensory system capable of taking measurements of the relative position between the landmarks and the robot itself. There are several approaches, the main ones being laser, sonar or vision based. Additionally, additional sensors such as compasses, infrared or GPS can be used to improve the perception of the state of the robot and the environment. [Aulinas et al., 2008] In navigation, odometry is the use of motion data from sensors, measured e.g. from encoders, to estimate the change in position and orientation over time. In the field of robotics and machine vision the above concept applies to inertial navigation, whereas if the position and orientation of the robot is determined by analysis of camera images the technique is known as visual odometry, and is specific to visual navigation methods. The ultimate goal of the SLAM process is to use the environment to update the robot's position. The robot's own odometry gives an approximate measure of the robot's position, but this is inaccurate due to sensor errors, so scans of the environment are often used to correct the position obtained, which is achieved by feature extraction from the environment and re-observation. Extended Kalman Filter (EKF) is the essential part of SLAM as it is responsible for updating the estimated position of the robot using the extracted features, known as landmarks. A schematic of the SLAM process can be seen in Figure 2.1.

When the odometry changes due to the robot's motion, the uncertainty of its new position is updated in the EKF by odometry update. Known points are extracted from the environment from the robot's new position and the robot tries to associate them with previously made observations. These landmarks are then used to update the robot's position in the EKF [Riisgaard and Blas, 2005]..

Figure 2.2 shows the initial state of a 2D SLAM process, with a mobile robot and 3 landmarks. Then, the robot is moved as shown in Figure 2.3. Once the movement is done, the odometry gives us an estimated position of the robot, and the distance to the points of interest is measured, according

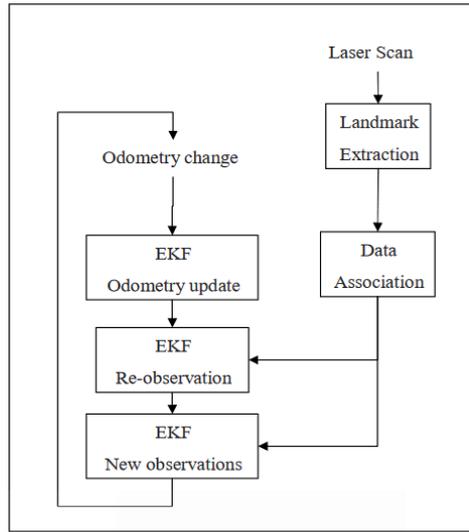


Figure 2.1: SLAM Scheme. Extracted from [Riisgaard and Blas, 2005]

to Figure 2.4. As the measurement given by the sensors is more reliable than their odometry, the measured distance to the landmarks is used to correct their position, as shown in Figure 2.5. Figure 2.6 shows the result of the method, where it can be seen that, although the sensors are not perfect either, they give a more accurate measurement of position than odometry.

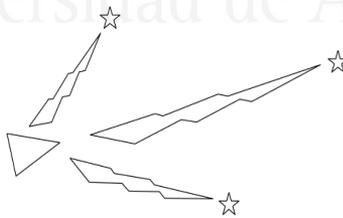


Figure 2.2: Initial status of robot. Extracted from [Riisgaard and Blas, 2005]

2.5.2 Registration of a scene

To generate a dense map with point clouds that can be used to represent, for example, the location of three-dimensional objects in the environ-

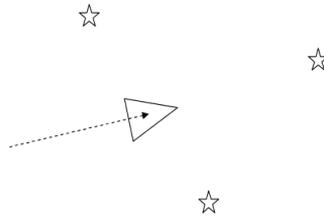


Figure 2.3: State of the robot after a movement. Extracted from [Riisgaard and Blas, 2005]

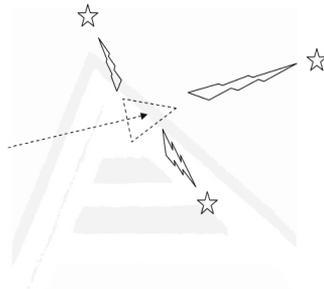


Figure 2.4: Measurement of distance to points of interest. Extracted from [Riisgaard and Blas, 2005]

Universitat d'Alacant
Universidad de Alicante

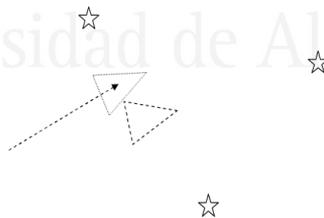


Figure 2.5: Correction of the robot's position after using the information from the points of interest. Extracted from [Riisgaard and Blas, 2005]

ment, the scene registration could be done directly by using the transformations calculated with the robot's position changes in SLAM and taking them to the reference system of the camera that performs the 3D scan of the environment and applying them directly to the point clouds obtained.

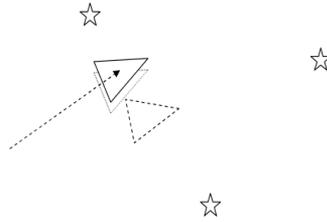


Figure 2.6: Final view showing the actual position of the robot (in bold), the position determined by the landmarks (dotted) and the position initially calculated by the odometry (dashed). Extracted from [Riisgaard and Blas, 2005]

However, as we have seen in the 2D SLAM example, the sensors are not entirely accurate, so there are techniques based on visual registration that try to match the 3D captures as closely as possible.

The most common method of scene registration is pairwise registration, which consists of calculating the transformations between two consecutive point clouds and accumulating them to generate a complete mapping, as shown in Figure 2.7.

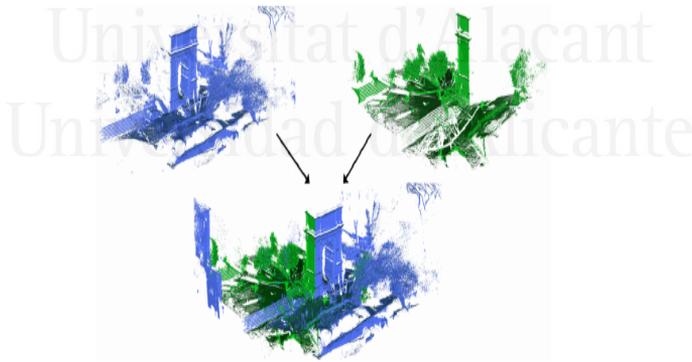


Figure 2.7: Example of pairwise registration of laser scans. Extracted from [ETH-Zurich, 2014]

The steps to be followed to carry out this technique are summarised in Figure 2.8 and are as follows:

1. **Data acquisition.** Obtaining point clouds from a 3D sensor.

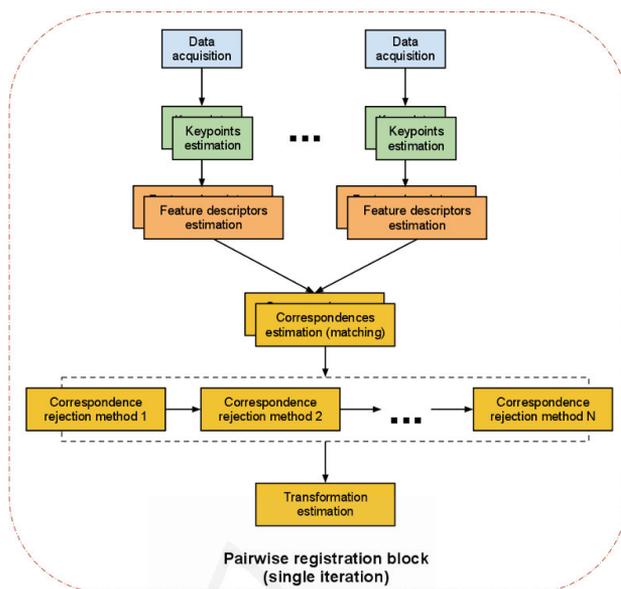


Figure 2.8: Pairwise point cloud registration pipeline. Extracted from [Point-Clouds, 2014]

2. **Keypoint estimation.** Extract distinguishable points of interest from the point cloud. The characteristics of a good *keypoint* are:

- Repeatable. There should be a good chance that the same points will be chosen in different iterations, even if the scene is captured from different angles.
- Distinguishable. The chosen points must be easily distinguishable and characterised for subsequent matching.

One of the most common keypoint detection algorithms is Scale Invariant Feature Transform (SIFT) 3D [Scovanner et al., 2007], the 3D version of the original algorithm of the same name [Lowe, 1999].

3. **Estimation of feature descriptors.** Descriptors are complex and precise signatures of a point that encode a large amount of information about its geometry. Their purpose is to uniquely identify a point regardless of the noise, resolution or transformation of the point cloud to which it belongs. The criteria for a feature to be optimal are as follows:

- Robust against transformations. Rigid transformations - those that do not change the distance between points - such as translations or rotations should not affect it.
- Robust against noise. Measurement errors should not change your estimate too much.
- Resolution invariant. If the points are sampled with a different density, the result should be identical or similar.

One of the most commonly used descriptors is Point Feature Histogram (PFH), which tries to capture information about the geometry of the point by analysing the difference between the directions of the normals in the neighbourhood, and its enhanced version Fast Point Feature Histogram (FPFH). [Rusu et al., 2009].

4. **Correspondence estimation.** Once the descriptors are computed for both point clouds, they must be matched to obtain point-to-point correspondences, usually using the Euclidean distance between vectors as the distance measure. To speed up the normal matching process, matching approximation schemes have been used, as in the case of Fast Library Approximate Nearest Neighbor (FLANN) [Muja and Lowe, 2014].
5. **Correspondence rejection.** As a result of the matching phase, the point-to-point correspondences are determined by associating pairs of descriptors of the two close point clouds in the descriptor space. Once this step is done, it is common to perform an additional step, consisting of discarding correspondences based on the geometric consistency between them. Assuming that the transformation between clouds is rigid, the set of correspondences for each cloud is grouped into subsets, each of which is responsible for the consensus for a specific rotation and translation of the model in the scene. Subsets whose consensus is too small are discarded, using a thresholding based on their cardinality [Zitova and Flusser, 2003]. In addition, it is common to use a threshold to discard correspondences that are unclear. One such technique is to extract the 2 nearest neighbours of

a point and reject the correspondence if the first nearest neighbour does not differ from the next by a certain percentage of the distance of the second.

- 6. Estimation of transformation.** The pre-validation step, while effective in discarding a large number of geometrically inconsistent matches, does not guarantee that all remaining matches in each cluster are consistent with a single rigid 3D rotation and translation transformation of the model on the scene. For this reason, it is useful to include an additional step, Random Sample Consensus (RANSAC), whose objective is to reduce the number of matches for each cluster inconsistent with the same 6 degrees of freedom pose. [Fischler and Bolles, 1981]. The model from which the consensus estimator is iteratively constructed is usually obtained by averaging an absolute orientation algorithm. Given a set of correspondences between the two point clouds, RANSAC in its 3D version determines the translation and rotation matrix that defines the rigid transformation that best explains it [Papazov and Burschka, 2011].

The generation of the 3D map is performed incrementally by pairs of consecutive point clouds. Once the transformation for a point cloud has been obtained to be joined with its predecessor, it only remains to apply that transformation matrix and the total transformation accumulated until then on the new point cloud, to enlarge the map of the environment.

2.6 Social robotics for personal assistance

The growing number of elder people is being increasingly important to promote the role and technological advance of social and assistive robotics. Speaking about Social Robotics creates the necessity to actually define what a Social Robot is. According to [Duffy et al., 1999], it is defined as *“A physical entity embodied in a complex, dynamic, and social environment sufficiently empowered to behave in a manner conducive to its own goals and those of its community”*.

In this specific field, we have to divide the different works based on their application. There are projects for assistance in medical environments [Ikeda et al., 2006] [Ikeda et al., 2005] whilst some others are focused on the emotional and cognitive tasks [Moyle et al., 2016] [Kouroupetroglou et al., 2017], and there also are projects for a social assistive task in different environments. This state of art is focused on this last issue.

In 1998 PAM-AID (Personal Adaptive Mobility Aid) [Rentschler et al., 2003] was created. This system aims to provide both physical support during walking and obstacle avoidance. It used sonar, infrared proximity sensors and bumpers switches to get information from the environment.

One year later the system proposed in [Van der Loos et al., 1999] emerged. It described the implementation of a control architecture for robots designed to combine a manipulation task with a motion controller that used the Operational Space formulation to define and implement arm trajectories and object manipulation.

The same year appeared the first so-called intelligent wheelchair [Hsu et al., 2012]. The system provided different functions: from fully autonomous navigation in an unknown crowded environment to partially autonomous local maneuvers. Two years later, on the same topic [Hillman et al., 2002] was created, which described the mounting of a robotic arm to a powered wheelchair to assist disabled users in daily activities.

In 2003 appeared three different types of work. First, [Falcone et al., 2003] describes the efforts to design, prototype and test a low-cost, highly competent personal rover for the domestic environment. Then, [Pineau et al., 2003] describes a mobile robotic assistant, developed to assist elderly individuals with mild cognitive and physical impairments, as well as support nurses in their daily activities. They used three software modules: an automated reminder system; people-tracking and detection system; and a high-level robot controller. Finally, [Pollack et al., 2003] uses AI techniques to model an individual's daily plans, observe and reason about the execution, and make decisions about whether and when it is most appropriate to issue reminders.

Years later, it appeared the PAMM [Karlsson et al., 1997] project, which is a system for support and guidance. The PAMM detects and

maneuvers away from obstacles, it uses an upward looking camera for localization and also can communicate with a central computer. The central computer provides the system with a map of the facility including the location. In turn, the system provides the central computer with the user's location, health status, and requests.

In 2010, the system described in [Fasola and Mataric, 2010] was proposed. It aims to design a socially assistive robot to monitor the performance of the user during a seated arm exercise scenario, the main purpose was to provide motivation to the user to complete the task and to improve performance.

Also this year, "the Home Exploring Robotic Butler" (HERB) [Srinivasa et al., 2010] was published. It can efficiently perform mapping tasks, searching, and navigation through indoor environments, recognize and localize several common household objects, and perform complex manipulation tasks.

In 2011, the system described in [Wang et al., 2011] appeared. It was an indoor mobile robot for taking care of the elderly. It has a human physiological parameters monitor system, which can take care up to six nursed persons by using a variety of sensors.

The ASIBOT [Huete et al., 2012] was published one year later. It helps users to perform a variety of tasks in common living environments. The robot is able to autonomously climb from one surface to another, fixing itself to the best place to perform each task. It also can be attached to a wheelchair, giving the user the possibility to move along with it as a bundle.

A new iteration of the aforementioned HERB [Srinivasa et al., 2010] system emerged in 2012. The HERB 2.0 [Srinivasa et al., 2012] consists of a two-handed mobile manipulator that can perform useful tasks for and with people in human environments.

In 2014, it was created a multi-user human-robot interaction (HRI) [Louie et al., 2014] system architecture to allow the social robot Tangy to autonomously plan, schedule and facilitate multi-user activities that considers the users' necessities. During the activities, the robot was able to interact with a group of users providing group-based and individualized

assistance based on the needs of the individual. The same year it was created the Robotic Nursing Assistant (RoNA) [Ding et al., 2014], which is able to assist nurses while performing intensive tasks and preventing musculoskeletal injuries among health care workers.

The Robotic Exercise Tutor [Görer et al., 2017] was published in 2017. A humanoid robot able to learn exercise routines from a human trainer and performs them in front of the elderly was created. Its main task was to monitor the performance of the patients and provide feedback.

In 2018, Deep Learning (DL) algorithms were introduced into assistive robot systems. For instance, a remote health-care system based on moving robots intended for the elderly at home was proposed in [Zhou et al., 2018a]. The robot is able to perform different kind of tasks. The user can control the robot and call it using voice commands or with a phone and it also performs object detection and pose estimation. It even can monitor the posture of the elderly and collect and transmit the data recorded by a set of sensors connected to the robot to the cloud for further analysis. On its behalf, PHAROS, a robotic assistant for assisting elderly in their daily physical activities at home, was proposed in [Costa et al., 2018a]. This interactive robot platform is divided into two modules: the Recommender (recommends activities at a scheduled time) and the Human Exercise Recogniser (as its name implies, it is the identifier of the human pose). This system works in real time and uses Deep Learning methods to properly recognize the performed physical exercises.

2.7 Social robotics for rehabilitation

The increase of disabled people together with a growing demand for rehabilitation worldwide require new ways to reduce the cost of the rehabilitation process while maintaining its quality. In this context, assistive technologies play an important role in functioning and increasing independence and participation [World Health Organization, 2010]. However, literature has mainly focused on the physical therapy.

In this line, Robotics has been a very active research area, going from companion robots for therapy to social assistive robots [Martinez-Martin

and del Pobil, 2017a]. So, rehabilitation robots are designed to assist people recovery in two different scenarios. On the one hand, robot systems can be used as a support tool in the rehabilitation process. This is the case of the Lokomat [Hocoma, 2018], an exoskeletal robot for physiological gait rehabilitation; NeReBot (NEuro REhabilitation roBOT) [Stefano et al., 2014], a cable-suspended device for upper limb rehabilitation of post-stroke patients; or, AMADEO [Tyromotion, 2018], a neurological rehabilitation device designed for the rehabilitation of the hand, fingers and thumb.

On the other hand, social autonomous robots could supervise the rehabilitation at home, a treatment appropriate for both people who cannot travel easily and those who require less care. From this starting point, for instance, Gomez-Donoso et al. [Gomez-Donoso et al., 2017b] developed a multisensor system for rehabilitation and interaction with persons with motor and cognitive disabilities. More recently, Costa et al. [Costa et al., 2018a] proposed PHAROS, an interactive robot system that recommends and monitors physical exercises at home designed for staying active, an important part of rehabilitation for chronic diseases.

Nonetheless, robotic solutions are far from being affordable, particularly in some low- and middle- income countries. In this regard, a virtual environment may be a reasonable substitute as shown in [Levy-Tzedek et al., 2017]. Actually, Virtual Reality (VR) has been used for evaluating and treating a number of pathologies. We have to note that the term VR nowadays implies the use of immersive technologies (like

So, Jack et al. presented a virtual reality system for rehabilitating hand function in stroke patients [Jack et al., 2001]. Four rehabilitation routines for hand recovery could be carried out: range, speed, fractionation or strength. For that, two different input devices must be worn: a CyberGlove and a Rutgers Master II-ND (RMII) force feedback glove.

Steffen et al. proposed a home-based platform for physical activity supervision and motivation [Steffen et al., 2013]. In particular, its system PAMAP (Physical Activity Monitoring for Aging People) uses the television as an interface for two applications: to set the exercises to be done (defined by a healthcare professional) and to provide feedback to the person. It is noteworthy that a group of sensors capturing person's motion is

required to be able to provide any information about their performance. Given that those sensors must be worn in strategic body positions, the person's performance could not be properly measured. In addition, a training phase is required to adjust the exercise performance evaluation to the person's physical limitations.

A more complex system is proposed by Toyra [Indra, 2019], a rehabilitation platform integrating healthcare information technology, virtual reality and motion capture to develop tailored interactive therapy exercises for upper limb recovery. As previously, several sensors should be worn to properly measure the patient's motion.

Virtual games can also be used in this context. Among them, Wii Fit [Nintendo, 2008] is highlighted since it has been applied in several scenarios such as Parkinson's disease [dos Santos Mendesa et al., 2012], knee replacement [Fung et al., 2012], stroke rehabilitation [Barcala et al., 2013] or balance recovery [Meldrum et al., 2012].

Another technology widely studied in the rehabilitation field is Augmented Reality (AR). In fact, AR provides the user with a better sense of presence and reality judgements of the environment as the interaction elements are real [Al-Issa et al., 2013]. This is the underlying idea of NeuroR [de Assis et al., 2014], an AR system for motor rehabilitation of chronic stroke patients that replaces the paralyzed arm in a virtual avatar. However, this kind of systems rely on privative technologies, such as the devices Apple Iphone or Apple Ipad that come with the Apple ARKit (Augmented Reality Kit), specialized hardware like Simblee, bio-signal sensors such as electrocardiography (ECG) or electromyography (EMG) or positional sensors such as inertial measurement unit (IMU). This is the case of the AR systems for upper-limb rehabilitation presented by Aung et al., i.e. ARIS (Augmented Reality based Illusion System) [Aung et al., 2014] and RehaBio [Aung and Al-Jumaily, 2014]. These systems combine a visual illusory environment with EMG signal (electromyography signal) to monitor the user's performance. Similarly, [Gazzoni and Cerone, 2018] used AR together with surface EMG (sEMG) detection/acquisition systems for physical rehabilitation. More recently, Monge & Postolache [Monge and Postolache, 2018] combined AR serious games with a wearable sensor net-

work based on *Simblee*, an Arduino-based programmable board with a wearable design, for physical rehabilitation of lower limb. Although the experimental results are very promising, the system requires an iPhone, what is not affordable for all the people.

Alternatively, SleeveAR [Sousa et al., 2016] is an AR system to perform rehabilitation exercises at home to complement in-clinic physical therapy of an injured arm. For that, three different stages take place: *the Recording Stage*, involves the demonstration of the exercise being recorded by the therapist; *the Movement Guidance Stage*, focuses on guiding the patient to recreate the prescribed exercise as previously recorded; and, *the Performance Review Stage*, provides the patient with an overview of their performance, by comparing with the original prescribed exercise. Note that this is a stationary system that requires a special covering floor and a custom sleeve, what considerably restricts its use. On its behalf, Desai et al. [Desai et al., 2016] used low-cost RGB-D cameras (i.e. Microsoft Kinect) to place the user into an AR scene for exercising by using Mirror therapy. Their pilot experiments result in a positive reinforcement. However, the set of exercises is very reduced (only 4 exercises) what considerably restricts its use. In a similar way, SilverFit [SilverFit, 2019] employs a 3D camera to register the user's movements and links them to a game. At the end, the game progress and the final results are displayed on the screen. It could be said that this system is the most similar to our approach since visual data is used to analyze the user's performance that can be recorded and reviewed afterwards. Despite the user progress and score is shown, there is no way to know the errors made except with a session with the therapist.

Note that all the proposed approaches have three main handicaps. The first one is the used of worn devices, what may considerably affect the patient's evaluation apart from the inconvenience caused. The second one is their cost what makes them unreachable for most of the disabled people. Thus, with the aim of overcoming these issues, we present a low-cost augmented reality system to monitor and evaluate physical rehabilitation at home. Finally, they do not include any *reference person* showing them how to do a specific exercise or a precise pose.

2.8 People re-identification

People re-identification in videos is a challenging problem but it also promises a huge potential for a wide range of applications mainly related with security and surveillance and health care or human-machine interaction [Gong et al., 2014].

An automated re-identification mechanism takes as input either tracks or bounding boxes containing segmented images of an individual person, as generated by a localized tracking or detection process of a visual surveillance system. To automatically match people at different locations over time, a re-identification process typically takes the following steps:

1. Extracting imagery features.
2. Constructing a descriptor or representation capable of both describing and discriminating individuals.
3. Matching specified probe images or tracks against a gallery of people in another camera view by measuring the similarity between the images.

A classic taxonomy classifies recognition methods as either single-shot when only one image pair is used, or multi-shot when two sets of images are employed. With regard to the learning approach, it is categorized as a supervised method if, prior to application, it exploits labeled samples for tuning model parameters. Otherwise a method is considered as an unsupervised approach and no training data is used to train the system.

In recent years, DL techniques have surpassed the classic methods in most computer vision challenges [Lavi et al., 2018, Yu et al., 2018, Wang et al., 2018a]. Furthermore, in [Qian et al., 2019] a multiscale re-identification system is proposed.

However, these models suffer from a lack of training data samples. This is because most of the available datasets provide only two images per individual [Gray and Tao, 2008], which makes the model fail at test time due to overfitting.

In this line, a number of new datasets have been proposed to solve this problem. Some of these are based on images: Market1501 [Zheng et al.,

2015], CUHK03 [Li et al., 2014], DukeMTMC-reID [Zheng et al., 2017], while other are based on video: MARS [Zheng et al., 2016], iLIDS-VID [Wang et al., 2014] or PRID2011 [Hirzer et al., 2011].

Some recent works using DL models include [Ahmed et al., 2015], which proposes a deep convolutional architecture with layers specially designed to address the problem of re-identification.

In [Chen et al., 2017], the authors learn multi-scale person appearance features using CNN by aiming to jointly learn discriminative scale-specific features and maximize multi-scale feature fusion selections in image pyramid input. In [Li et al., 2018c], a Tracklet Association Unsupervised Deep Learning (TAUDL) framework is proposed. It is characterized by jointly learning per-camera (within-camera) tracklet association (labeling) and cross-camera tracklet correlation by maximizing the discovery of most likely tracklet relationships across camera views. Some approaches employ graph deep neural networks like [Shen et al., 2018], which proposes a novel DL framework called Similarity-Guided Graph Neural Network (SGGNN). Given a probe image and several gallery images, SGGNN creates a graph to represent the pairwise relationships between probe gallery pairs (nodes) and uses such relationships to update the probe gallery relation features in an end-to-end manner.

A comprehensive and exhaustive survey on person re-identification can be found in [Wang et al., 2018b]. In this analysis, it is concluded that the main drawback of DL-based approaches is that they cannot assure high accuracy and low computation cost because they need constant retraining.



Universitat d'Alacant
Universidad de Alicante

3D Object Recognition

In this chapter, we describe the contributions made within the field of 3D object recognition. A brief introduction is provided in Section 3.1. Section 3.2 presents a novel 3D object recognition method based on local similarities of free-form shapes, more specifically using Nurbs. Finally, Section 3.3 explains a new method for 3D object recognition using the fractal dimension as a descriptor.

3.1 Introduction

The ability to recognize tridimensional objects is a key feature for a range of different applications. For instance, intelligent and self-driving cars must be able to detect and recognize the objects that surround them in order to enable a specific behaviour: they must stop in a red light, they must engage the emergency brake if a pedestrian suddenly crossed the road or they must recognize the traffic signs to drive safely.

Another notorious application for the tridimensional object recognition could be social robotics. The robots that are meant to interact with humans should be able to recognize each of them, and also to recognize certain objects that may be potentially involved in their tasks. For instance, it might be useful for the robot to recognize household items [Naseer et al., 2019].

Until 2012, all the proposals submitted to the *Large Scale Visual Recognition Challenge*, an object recognition challenge with ImageNet images

database [Russakovsky et al., 2015], were based on handcrafted features and classical classifiers. However, that year appeared the first *Deep Learning* approach [Krizhevsky et al., 2012], that outperformed the previous techniques.

Since that time, the research focus changed to *Deep Learning*, with a great and rapid evolution and an enormous set of new architectures and learning strategies. For the 2D object recognition task, several approaches have achieved high classification rates, as explained in the survey [Liu et al., 2018a]. However, 3D object recognition is still an underexplored research field compared to its counterpart in 2D.

So far, the most novel and accurate 3D object detectors rely in a deep learning pipeline. However, these methods require a large amount of annotated and structured data in order to be trained, which is very hard to obtain. Due to this, most of the deep learning-based 3D object detectors do not perform that well in real life, off-lab tests. In addition to that, most of them are coupled to a certain kind of sensor. For instance, there are approaches that only work on 16-Beams Laser Imaging Detection and Ranging (LiDAR) data, or they only work on 3D Computer Aided Designed (CAD) objects.

In the following sections, we will present two novel and alternative methods for 3D object recognition, based on concepts such as surface similarity and fractal dimension.

3.2 Object recognition using free form surfaces

In this section, we propose *NurbsNet*, a novel approach for 3D object recognition based on the search of local similarities between the object and internal free form surfaces.

3.2.1 Approach

Our method receives a point cloud as input and generates a probability distribution of labels for the received object. Our proposal introduces two new approaches: a *Nurbs layer* that learns free form surfaces and calculates the similarity between every Nurbs and local shapes of the input cloud,

and a *Voxelization layer* that uses a spherical grid that selects the best activations and generates a feature vector to feed the *Fully Connected layer*.

In Section 3.2.2 we explain the fundamentals of the Nurbs surfaces, the fitting process and the similarity calculation. In Section 3.2.3, we present the architecture of our proposal with an extensive explanation of our novel layers.

3.2.2 Nurbs surfaces

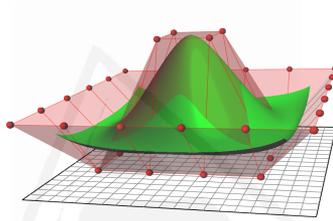


Figure 3.1: Representation of a Nurbs surface. Extracted from [Chrschn, 2019]

We have chosen Nurbs to represent freeform surfaces in a parametric form. This mathematical model has been widely used by the CAD industry and the academic community as a 3D representation due to its expressiveness and relative simplicity. This kind of representation allows modifying the local geometry of an object by moving a few control points, so it favors surface optimization.

A Nurbs surface is represented by Equation 3.1, where $P_{i,j}$ are the 3D control points, $w_{i,j}$ the weights for every control point and $N_{p,q}$ are the normalized B-spline basis functions of degree p . Every $N_{p,q}$ function affects the surface in a limited range, defined by the knot vectors [Piegl, 1991].

$$S(u, v) = \sum_{i=1}^k \sum_{j=1}^l R_{i,j}(u, v) \mathbf{P}_{i,j} \quad (3.1)$$

$$R_{i,j}(u, v) = \frac{N_{i,n}(u) N_{j,m}(v) w_{i,j}}{\sum_{p=1}^k \sum_{q=1}^l N_{p,n}(u) N_{q,m}(v) w_{p,q}}$$

3.2.2.1 Nurbs fitting

Nurbs fitting consists of the reconstruction of a complete surface from a limited set of 3D data, calculating its parameters iteratively.

The simplest method to minimize the distance between the pointcloud and the generated surface is *least mean squares minimization*. The measure for the distance can be the Euclidean distance, known as *point-distance minimization* (PDM), the *tangent distance minimization* (TDM) [Blake and Isard, 1998] or the *squared distance minimization* (SDM) [Wang et al., 2006].

The surface is initialized using *Principal Component Analysis* (PCA), using the plane formed by the two eigenvectors with the greatest eigenvalues of the data, where the initial control points will lie. Then, in every iteration the distance between the points and the surface is calculated according to some distance criteria (as mentioned above) and the control points are updated in the direction where the distance decreases.

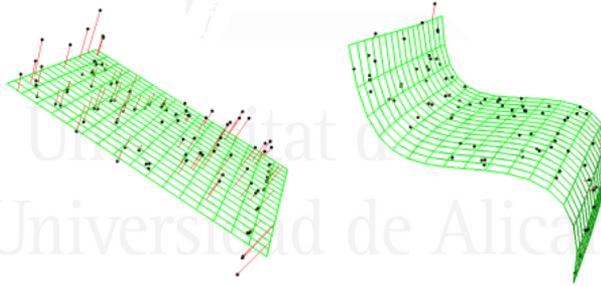


Figure 3.2: Least mean squares minimization of the distance between the points and the surface. Extracted from [Moerwald, 2019]

The fitting method was proposed in [Piegl and Tiller, 1997] and the implementation has been taken from the *Point Cloud Library*, carried out by Thomas Moerwald [Moerwald, 2019].

3.2.2.2 Similarity metric between two Nurbs surfaces

There are some works like [Liang et al., 2003] and [Liang et al., 2002] that propose a similarity measure of Nurbs using the Nurbs Warping method [Piegl, 1991] for image retrieval purposes. However, these tech-

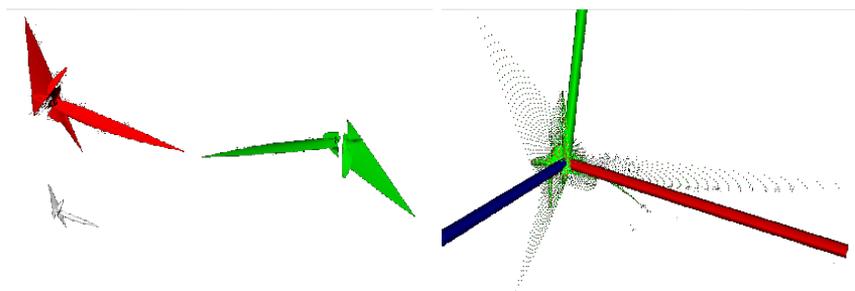


Figure 3.3: Red, green and grey nurbs surfaces in the top have the same shape but different scale, rotation and translation. They are represented in the bottom with the same sample points in euclidean space after the use of the techniques explained in Section 3.2.2.2

niques are only applicable to 2D images because the shape of a 3D object is far more complex and cannot be approximated by simply *fitting their borders*.

To solve the similarity problem between Nurbs surfaces in 3D, we propose a pointwise approach, using the following method:

1. **Sampling of the surface.** Using the Nurbs equation and iterating over the parameters u and v we can obtain a set of points from the surface. There are some studies [Piegl and Tiller, 1999] and [Pagani and Scott, 2018] that investigate this topic in order to choose the proper parameters that generate the points that better represent the surface.
2. **Normalization.** We want identical surfaces at different scales to be able to have minimum distance, so we have to make the sample point representation invariant to scale. First, we normalize for translation by aligning the centroids of the two surfaces. Then we calculate the median distance of the points from the centroid. Then, we carry out the normalization step by dividing every coordinate by the median. Dividing by the maximum distance could be an alternative, but the median is less affected by noise.
3. **Principal Components Analysis.** Similar surfaces with equivalent but rotated principal axes should have lower distances. In order to achieve the rotation invariance, we do *Principal Component*

Analysis (PCA) on the pointcloud to obtain the axis that capture the maximum variability, and apply the transformation matrix that changes its basis, as explained in [Smith, 2002]. *PCA* does not ensure the orientation of the axis (only the direction), so we save the four possible permutations of the sign of two principal vectors, because the third axis is calculated via vector product. We will use the four representations of every surface to calculate the distance to another figure.

4. **Distance measuring.** We calculate the distance of every sampled point of the first surface to the other surface. This distance can be measured in several ways, with the euclidean distance to the nearest neighbour in the other surface the most common, simple, and good enough for our purposes, as shown in Equation 3.5. The resulting distance between the surfaces is the maximum of the two minimum unidirectional distances (Equation 3.3). Finally, the similarity is calculated from the resulting distance using Equation 3.2. Notice that in Equation 3.4, when we calculate the distance from one surface to another, we only iterate over the permutation of the axis of one of the surfaces to exploit the symmetries of the representations and reduce computational costs.

As the output of this process, we obtain a similarity score that will be 1 when the surfaces represent the same shape, and will decrease as their differences grow. In this way, we have represented the points of the Nurbs with scale, translation and rotational invariance.

$$\text{similarity}(s1, s2) = 1 - d(s1, s2) \quad (3.2)$$

$$d(s1, s2) = \max(d_unidir(s1, s2), d_unidir(s2, s1)) \quad (3.3)$$

$$d_unidir(s1, s2) = \min_{\forall i \in s1} d_sing(s1_i, s2_1) \quad (3.4)$$

$$d_sing(s1_i, s2_j) = \frac{1}{|s1_i|} \sum_{\forall p_i \in s1_i} d(p_i, n_neigh(p_i, s2_j)) \quad (3.5)$$

3.2.3 Network architecture

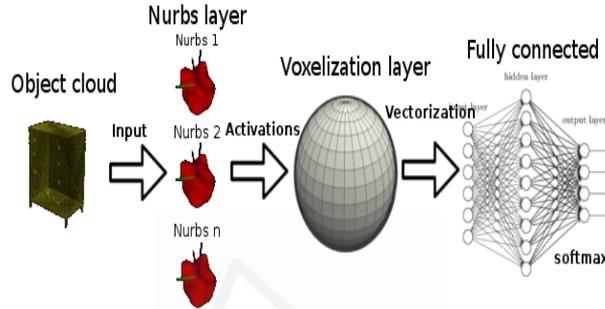


Figure 3.4: End-to-end network scheme of our proposal

The scheme of our end-to-end architecture is depicted in Figure 3.4. The *Nurbs layer* receives the point cloud of the segmented object. It calculates the activations for every point with every internal Nurbs in the layer and sends this information to the *Voxelization layer*. This layer discretize the space around the object with a spherical mesh divided in prism sections, and calculates the best activation for every Nurbs that lies in every sector. Finally, it flattens the output as a vector and sends it to the *Fully connected layer*, which calculates a classification probability for every class.

The *Nurbs layer* is explained in Section 3.2.3.1 and the *Voxelization layer* in Section 3.2.3.2.

3.2.3.1 Nurbs layer

This layer is directly connected with the pointcloud input of the network. As a preprocessing step, we perform a normalization of the cloud similar to the one explained in Section 3.2.2.2 for Nurbs. Additionally, we can apply a voxel grid filter to reduce the dimensionality and favour the generalization of the learning process.

The Nurbs layer contains a set of internal Nurbs and replaces the classical convolutional layer of a *CNNs* by calculating similarity scores between surfaces. It is important to notice that this layer can work with unordered point clouds of variable size.

The main task carried out by this layer is the calculation of the similarity score between the internal Nurbs and every local Nurbs of the input, following these steps:

1. **Fit local Nurbs for the input.** For every point in the input point cloud, extract its neighbors using a fixed radial neighbourhood. Then, apply the fitting process explained in Section 3.2.2.1. The result of this process is a local Nurbs for every point in the cloud.
2. **Calculate activations.** For every local Nurbs generated in the previous step, calculate their similarity score with all the internal Nurbs of this layer, as explained in Section 3.2.2.2.

This layer has some hyperparameters that must be set: the radius of search for neighbors, the number of internal Nurbs and their number of control points. Furthermore, there are some strategies for initializing the Nurbs control points: random initialization (using a normal distribution), initialization from geometric surfaces and initialization from real surfaces.

3.2.3.2 Voxelization layer

This layer receives the output from the *Nurbs layer*, explained in Section 3.2.3.1, as a vector of activations of every point in the point cloud with every internal Nurbs in the *Nurbs layer*. The main goal of this layer is to generate a descriptor that can be suitable for the input to the fully connected neural network. In this case, we have chosen a discretization technique of the space around the point cloud, but in a different manner than usual.

Many approaches like [Garcia-Garcia et al., 2016] and [Xu and Todorovic, 2016] build a volumetric occupation grid, in form of cube voxels, to represent the shape of the figure. However, point clouds are usually hollow in their center, so this technique is not efficient for our purposes as it produces so many holes in the final vector representation.

In this case, we propose another technique of discretization, based on a spherical coordinate space, following these steps:

1. **Convert points into spherical coordinates.** Calculate the centroid of the pointcloud and apply a translation to convert the centroid to the origin of coordinates (0,0,0). Convert the points from cartesian to spherical coordinates following Equation 3.6.

$$\begin{aligned} R &= \sqrt{x^2 + y^2 + z^2} & \phi &= \tan^{-1} \left(\frac{y}{x} \right) \\ \theta &= \cos^{-1} \left(\frac{z}{\sqrt{x^2 + y^2 + z^2}} \right) \end{aligned} \quad (3.6)$$

2. **Normalize.** Normalize the radius component R dividing it by the largest radius of the point cloud.
3. **Discretize.** Discretize every spherical coordinate using the defined hyperparameters of resolution for every coordinate. Radius R is defined in the range $[0, 1]$, θ in the range $[0, \pi)$ and ϕ in the range $[0, 2\pi)$. It's recommended to use a resolution of 1 for the radius component to minimize the holes of the representation. We divide the space into prism sections according to the provided resolution, similar to the techniques used in 3D descriptors like 3D Shape Contexts [Körtgen et al., 2003] or Unique Shape Context [Tombari et al., 2010].
4. **Select activations.** First, we identify which points lie in every prism section. Then, for every data point in that section and every internal Nurbs, we look for the best activation based on some criteria like maximum, minimum or average activation. This selection criteria does a similar task to the pooling layers of the *classical* convolutional networks.
5. **Build representation** Using the best activation of every Nurbs in every prism section, build a linear vector with these values given an order of sections.

As an output of this layer, we have a linear representation of the input data that represents the local similarities of the point cloud with the

internal Nurbs of the *Nurbs layer*. This vector of characteristics is fed to the fully connected layer to perform the classification of the object.

3.2.3.3 Fully connected layer

This layer receives the linearized output of the previous layer (vector of characteristics). It is a conventional fully connected layer that performs the classification part of the network. Consists in a set of neuron layers that are fully connected with the previous layer with a set of weights, which are learnt with backpropagation and gradient descent techniques, that apply nonlinear activation functions such ReLu, tanh or sigmoid. For classification purposes, the last layer of the fully connected is a logit layer that performs a softmax classification as the probability inference (Equation 3.7).

$$\text{softmax}_i(a) = \frac{e^{a_i}}{\sum_{\forall a_j \in a} e^{a_j}} \quad (3.7)$$

3.2.4 Experiments

In order to test and compare our architecture with other state-of-the-art methods, we have chosen the *Princeton ModelNet* dataset [Wu et al., 2015b], explained in Section 1.2.10.1.

The experiments have been carried out using our own Deep Learning framework, due to the novelty of our proposed method and the use of C++ third party libraries (*PCL*).

3.2.4.1 ModelNet10

This experiment has been carried out using a *Nurbs layer* with radius search of 0.10 (after cloud normalization) and 2 internal Nurbs, initialized randomly. In the voxelization layer, we have chosen a resolution of (1,30,30) for R , ϕ and θ respectively and a max pooling activation strategy. The fully connected layer consists of an input layer of 1800 neurons (1x30x30x2), an internal layer of 1800 neurons with a sigmoid activation and a final softmax layer with 10 outputs. The learning rate has been set to 0.01, adaptive, and the batch size is 10.

		Predicted									
		bathhtub	bed	chair	desk	dresser	monitor	nightstand	sofa	table	toilet
Actual	bathhtub	80	12	0	0	0	0	0	6	2	0
	bed	0	98	1	0	0	0	0	0	1	0
	chair	0	2	96	0	0	0	0	0	2	0
	desk	0	0	1	77	2	1	1	8	9	0
	dresser	0	0	0	1	72	4	23	0	0	0
	monitor	0	0	2	0	1	93	1	1	2	0
	nightstand	0	0	0	2	21	1	71	0	5	0
	sofa	0	0	0	1	1	0	1	97	0	0
	table	0	1	0	17	0	0	0	0	82	0
	toilet	0	1	4	0	0	0	1	0	0	94

Figure 3.5: Confusion matrix of the classification test results achieved by *Nurb-snet* after 50 training iterations using the *ModelNet10* dataset with 2 Nurbs. The values shown in the table are expressed as percentages. It is remarkable that there is some confusion between the pair of classes desk-table and dresser-nightstand, which are very similar.

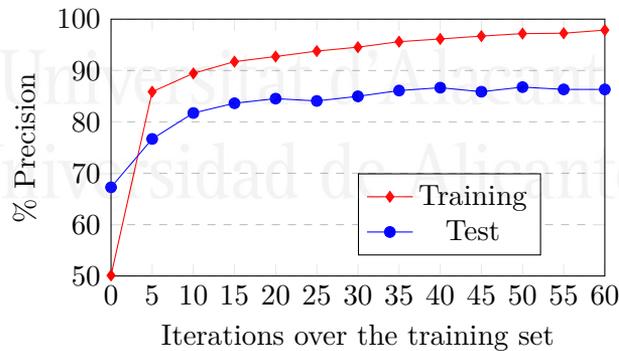


Figure 3.6: Training and test precision achieved by *Nurb-snet* after 60 training iterations using the *ModelNet10* dataset with 2 internal Nurbs

The final classification precision for test set is 86.77%. Despite the fact that this is a good result, this score is much more interesting if we analyse the confusion matrix provided in Figure 3.5. We can see that similar objects such as the pairs desk-table and dresser-nightstand are getting confused, but this kind of confusion is typical even for humans, because

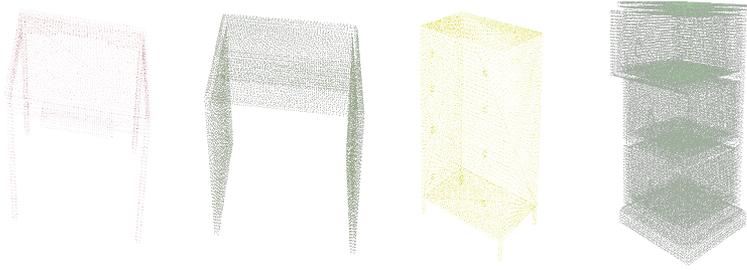


Figure 3.7: Pairs of confused objects. From left to right and top to down: desk classified as table, table classified as desk, dresser classified as nightstand and nightstand classified as dresser.

the difference is more semantic (the use that we make of these objects) than on their shapes. In Figure 3.7 we see an example of these point cloud pairs. If we count the misclassification between these pairs as a hit, the real precision would be 94.62%, which is similar to state-of-the-art methods, as shown in the table presented in [Wu et al., 2019], with much less iterations over the entire training dataset than another methods based on convolutions and only 6480000 parameters, which yields to a minimum memory consumption of 200 MB.

3.2.4.2 ModelNet40

This experiment has been carried out using a *Nurbs layer* with radius search of 0.10 (after cloud normalization) and 2 internal Nurbs, initialized randomly. In the voxelization layer, we have chosen a resolution of (1,30,30) for R , ϕ and θ respectively and a max pooling activation strategy. The fully connected layer consists of an input layer of 1800 neurons (1x30x30x2), an internal layer of 1800 neurons with a sigmoid activation and a final softmax layer with 40 outputs. The learning rate has been set to 0.01, adaptive, and the batch size is 10.

The final classification precision for test set is 75.13%. Despite the fact that this is not a bad result at all for 40 classes, this score is much more interesting if we analyse the confusion matrix provided in Figure 3.8. We can see that there are also new similar pairs like cup-vase, flower pot-vase, flower pot-plant and curtain-door that are quite similar in many cases in

the dataset and are confused. In Figure 3.10 we see an example of these point cloud pairs. If we count the misclassification between these pairs as a hit, the real precision would be 79.28%, which is about some of the state-of-the-art methods, with much less iterations over the entire training dataset and only 6480000 parameters, which yields to a minimum memory consumption of 200 MB, much lighter than other methods presented in [Wu et al., 2019].

3.2.5 Conclusions and future works

In this section, we proposed *NurbsNet*, a new approach for tridimensional object recognition based on the local similarities of the objects with a set of internal Nurbs surfaces. It provides a fast convergence method, with few parameters and memory consumption, and achieves very good precision results with just a few iterations over the entire dataset.

Although the current results do not outperform the state-of-the-art methods, we consider that this completely new approach has a lot of room for improvement and can inspire new different methods to handle the 3D recognition problem.

Moreover, the internal outputs of the *Nurbs layer* and *Voxelization layer* can be inspected and easily analysed, giving a first step towards the explainability of the decisions taken by the system, a hot topic at this moment.

Following on this work, we are planning to add more *Nurbs layers* with different values of radius search, even adaptive, in order to learn characteristics of different levels of complexity from the point clouds. Moreover, we will test this approach with partial views of the objects and occlusions, from real 3D sensors as the *Kinect*.

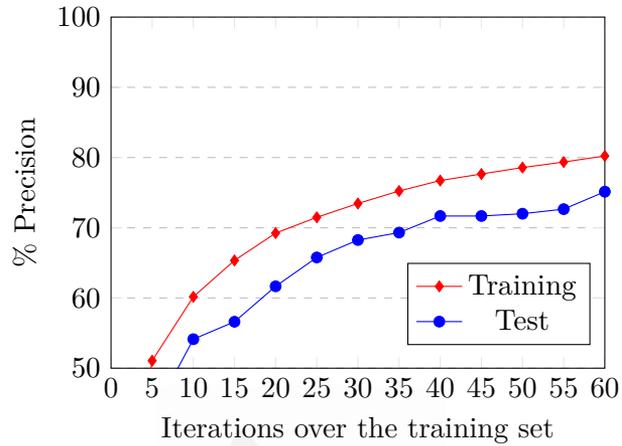


Figure 3.9: Training and test precision achieved by *Nurbsnet* after 60 training iterations using the *ModelNet40* dataset with 2 internal Nurbs



Figure 3.10: Pairs of confused objects. From left to right and top to down: cup classified as vase, vase classified as cup, flower pot classified as vase, reference of vase object, flower pot classified as plant, reference of plant object, curtain classified as door, door classified as curtain

3.3 Object recognition using fractals

In this section, we propose a novel global descriptor, which is based on the fractal dimension. The voxel-based computation of the fractal dimension is agnostic to the density of points, number of points in the input cloud, sensor of choice, and noise up to a level. In addition to that, it is very fast to compute. The fractal descriptor could be used for 3D object recognition, which is the feature that is most explored in this work.

Thus, the main contribution of this section is a novel global descriptor that is based on the computation of the fractal dimension, which we called Voxelized Fractal Descriptor (VFD).

3.3.1 Proposal

In order to evaluate the feasibility of the fractal dimension as a 3D object descriptor, we propose the VFD. The process to generate the VFD from an input point cloud is as follows: first, a point cloud is obtained. The points are not required to be sorted by any mean. A voxel grid clustering of a specific size is computed using the point cloud as an input. The resolution is a parameter of the approach. As a result, there are generated a set of clusters which contain the points in the input point cloud. Then, the fractal dimension is computed for each cluster. As a result, we obtain a set of fractal dimensions, one per cluster. If a cluster is empty, namely it has no points within, a fixed sentinel value of 0 is returned, corresponding with the theoretical fractal dimension of the empty set. Finally, the fractal dimensions are concatenated to generate the final voxelized fractal descriptor. As expected, the computational complexity of the descriptor is related to the resolution of the voxel grid filter, as more voxels implies the computation of more fractal dimensions, thus generating a larger descriptor. The process to generate the VFD is shown in Figure 3.11.

The explanation of the algorithm to calculate the fractal dimension is in Section 3.3.1.1, and the details of our descriptor are in Section 3.3.1.2

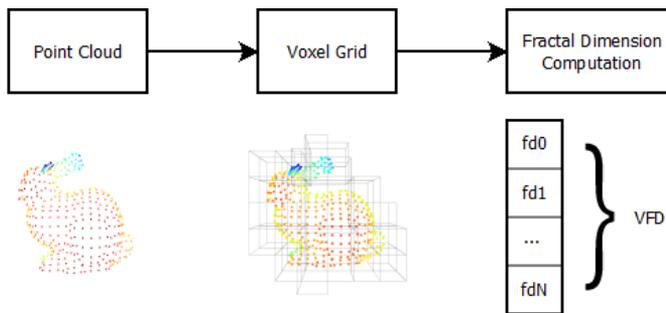


Figure 3.11: The process to generate a VFD from an input point cloud.

3.3.1.1 3D fractal dimension algorithm

The concept of fractal dimension, in fractal geometry, is a generalization of the concept of dimension in classic geometry. As a simplification, it can be defined as a measure of how much an object appears to fill the space as it is scaled up or down. A clear example is the Koch snowflake. This figure has a topological dimension of 1 but cannot be treated as a curve, because as we zoom in on the figure, the distance between its points tends to infinite. This phenomena is know as Coastline Paradox [Mandelbrot, 1993].

There are many ways to calculate this property (information dimension, correlation dimension, Higuchi dimension, Lyapunov dimension, Hausdorff dimension...), but we have chosen the box counting method, also known as Minkowski–Bouligand dimension. This method is one of the most popular to calculate the fractal dimension of sets in euclidean space, it is directly applicable to 3D data, and is very fast to compute.

The name of the box counting method is very descriptive, as it consists in dividing the space in boxes, and counting the number of cells that contain points. In more detail, the space occupied by the object is divided into square boxes of side size ε . Then, the cells that are occupied by the object are counted. This process is repeated $nIters$ times with a fixed *boxSizeDecrease* decreasing value of ε , obtaining the Equation 3.8, where $N(\varepsilon)$ is the number of boxes required to cover the figure according to the side size, ε is the side size, and D_{BC} is the box counting dimension.

$$N(\varepsilon) = \varepsilon^{D_{BC}} \quad (3.8)$$

The obtained expression is similar to those of Hausdorff and Higuchi, and can be operated with algorithms to generate a linear function, as shown in Equation 3.9. As we can see, D_{BC} is the slope of the linear function. In our practical application, we will apply a least-squares adjustment to approximate the value of this slope, as proposed in [Rian and Sassone, 2014].

$$\log(N(\varepsilon)) = D_{BC} \cdot \log(\varepsilon) \quad (3.9)$$

The translation of this algorithm to the 3D euclidean space is a trivial process, as the only difference is the substitution of the squares boxes of the grid by cubic voxels. To do so, we relied in the voxel grid algorithm. The only parameter to be set is the number of iterations $nIters$ (decreases in box size), as the rest of parameters are calculated from this. Assuming that the object is centered in the origin, the initial box size, tam_0 is the distance from the origin to the furthest point of the object, and the decrease in box size is calculated following the Equation 3.10

$$boxSizeDecrease = \frac{tam_0}{nIters} \quad (3.10)$$

The voxel grid filter is applied in each iteration, as shown in Figure 3.12 and Figure 3.13 with the corresponding size. Then, the number of voxels that are occupied is computed. We stored the corresponding values of voxel size and number of occupied cells in every iteration, and applied the Equation 3.9 approximating, thus, the slope of the line with a least-squares adjustment. Finally, this value represents the fractal dimension, as shown in Figure 3.14.

3.3.1.2 3D voxelized fractal descriptor

The one-dimensional version of the descriptor is explained first. Given a 3D object defined by a set of 3D points that represent its surface, this descriptor is calculated for the whole object using the method explained

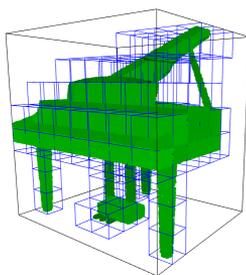


Figure 3.12: Sample of the division (in blue) of an instance of the *piano* class from *ModelNet40*. In black we can see the main bounding box of the figure, that marks the size for the divisions.

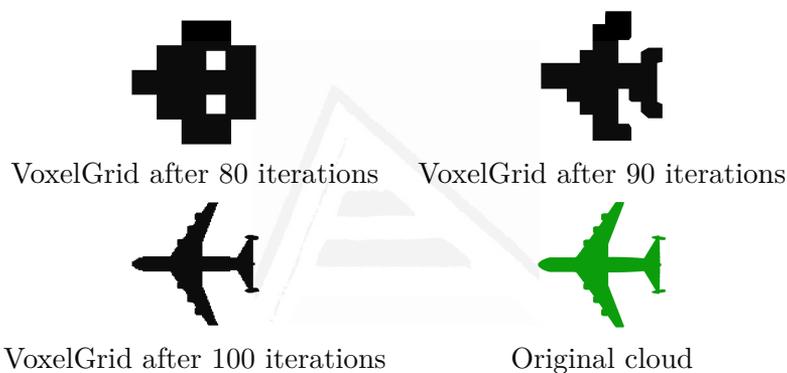


Figure 3.13: Effects of the evolution of Voxel Grid filter in box counting.

in the previous section. Thus, as a result, there is a single value which is used as a global descriptor for the object. In Section 3.3.2 we describe an experiment to check the efficacy of this method.

The next idea was to involve the distribution of points and their location on the object as well. With this in mind, our proposal is to divide the object in voxels, with a voxel resolution provided by the user. Then, the fractal dimension for the points that lie within each voxel is computed. If a voxel has no points within, a fractal dimension of 0 is returned. The fractal dimensions are concatenated in a single vector by following the order of creation, thus generating the final descriptor. We named this version the VFD. It is worth to mention that the only parameter that takes VFD is the resolution of the voxels. The descriptors of a set of objects are then used to train a classifier in order to tackle the 3D object recognition task.

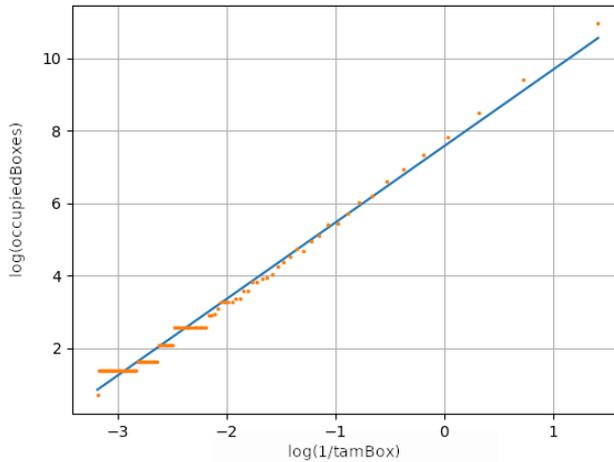


Figure 3.14: Example of least-squares approximation for fractal dimension.

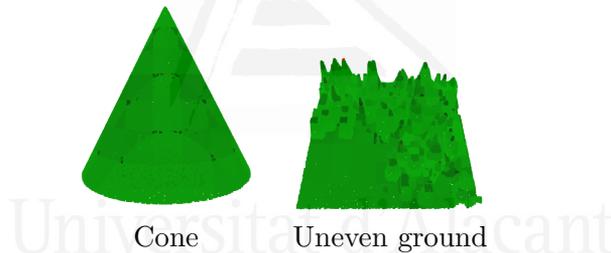


Figure 3.15: Plot of the calculation of fractal dimension over two voxelized figures, a cone and a ground with two different parts: a flat surface and a very sharp surface, with the lighter colours being higher values of fractal dimension

Figure 3.15 shows the result of dividing a cone in voxels and calculating the fractal dimension for every voxel. The points that lie in a certain voxel are colored as a function of the fractal dimension. The voxels with the lightest shades of greens, which represent a higher fractal dimension, correspond with the zones with more points. The darkest areas represent a lower fractal dimension, which also are zones with fewer points. On the other side, we can see the difference between a flat surface and a rough surface. Logically, for the regular part of the surface, the fractal dimension is constant, whilst the roughest parts show a range of different fractal dimension. In this part, the bulge tips have low values, the points

near the base have similar values to the flat surface, and the central parts present the highest values of fractal dimension, which are represented by the different shades of green.

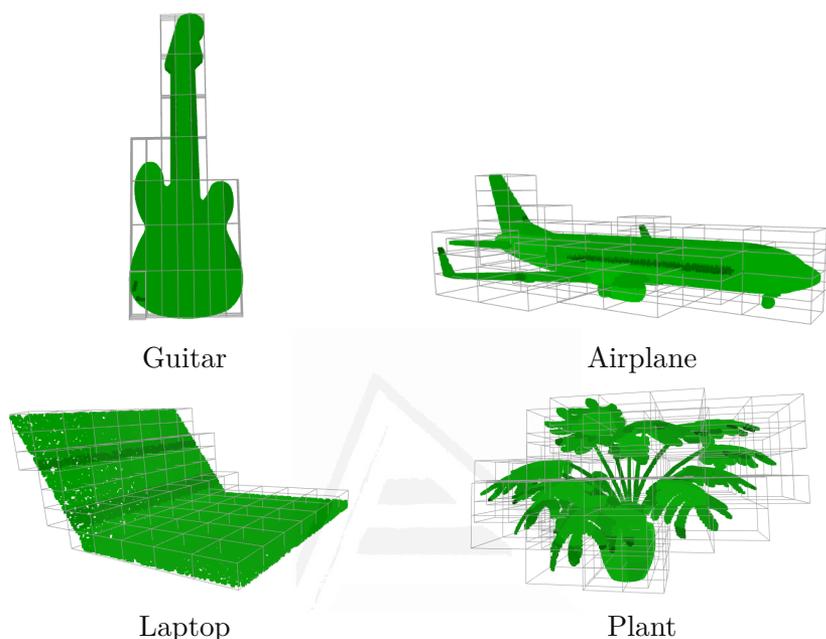


Figure 3.16: ModelNet40 samples of calculating fractal descriptor over voxels. Lighter green values are higher fractal dimensions.

In Figure 3.16 we present the analysis for 4 different classes of the ModelNet40 dataset. First, it shows a guitar, where its body and the central part of the neck are constant, because they are flat surfaces. The parts that show more difference are the edges of the guitar neck and body. Besides, there is a darker band due to the intersection of some voxels with the limits of the cylinder. More dark points are on the winglets, at the end of the plane. The case of the laptop is pretty interesting because the keyboard is in constant color as it is vertically aligned, but the screen shows some discontinuities due to the fact that it is tilted, thus its points lie within multiple columns of the grid. Finally, we have the example of a plant, which leaf surfaces and vase are of similar, light green color, whilst the stem and leaf tips are of darker colors. This effect is identical to the one with the wings of the plane, the guitar tuning pegs and the tips of

the rough surface. We strongly believe that this is an important factor to differentiate between objects, as we demonstrate in the next section.

3.3.2 Experimentation

In this section, we will carry out the experiments using ModelNet dataset, explained in Section 1.2.10.1. First, we apply the one-dimensional descriptor to the smallest version of the dataset, *ModelNet10*, and show the results in Section 3.3.2.1. After that, in Section 3.3.2.2, we present the results for the *ModelNet10* dataset for the Voxelized Fractal Descriptor. For this dataset, we conducted a great deal of tests in order to find the classifier and the voxel resolution that provide the best performance for our descriptor. Then, in Section 3.3.2.3 we use the best setup found in the previous section with the most complex version of the dataset, which is *ModelNet40*. We do this to evaluate the performance of our system with harder problems, also with the VFD descriptor.

The experiments were carried out using the following setup: Intel Core i5-3570 with 16 GiB of Kingston HyperX 1600 MHz and CL10 DDR3 RAM on an Asus P8H77-M PRO motherboard (Intel H77 chipset). The implementation of our methods has been made with Python, using the novel Open3D library [Zhou et al., 2018b], which offers a large set of functions to process 3D data.

3.3.2.1 Results on ModelNet10 using the 1D descriptor

The first experiment we carried out consists of a naive experiment, using the one-dimensional descriptor, which is described in Section 3.3.1.2, for the ModelNet10 dataset. We have used the K-Nearest Neighbors (KNN) classifier, with a K value of 5. It showed a very poor accuracy of 21,6%, as shown in Figure 3.17. This result is expectable as it is very likely that a range of different object share the same fractal dimension for the whole object. Therefore, we conclude that this descriptor does not provide sufficient discriminative power to allow us to distinguish between categories, so we need to divide the object into parts and calculate the fractal dimension for each of them.

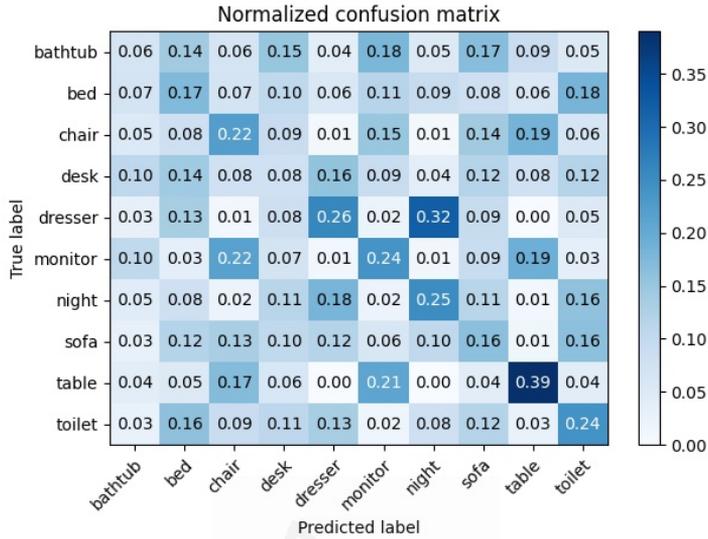


Figure 3.17: Confusion matrix with the one-dimensional fractal descriptor for the whole object with a KNN classifier, $k = 5$.

3.3.2.2 Results on ModelNet10 using VFD

The first classifier we tested was KNN. We tried different setups of k and voxel resolutions. The results are presented in Figure 3.18. In this plot, it can be seen the influence of the k parameter and the voxel resolution. With the minimum number of subdivisions (2^3) the worst results appear, but they improve considerably from 3^3 to 10^3 , where the accuracy starts decreasing slightly. Apparently, this algorithm works better with lower values of k , regardless of the voxel resolution. Moreover, it is worth noting that the number of subdivisions with better average result is 5^3 . Finally, the best result of a 86,01% accuracy is obtained with the configuration $k = 1$ and 5^3 subdivisions. The confusion matrix regarding this setup is shown in Figure 3.19. This last result will be used to compare with other classifiers.

The next classifier we tested was a fully-connected neural network. We tried a simple topology as a baseline in order to know the reliability of our descriptor and prevent overfitting. This first architecture has only two layers: an input layer with a number of neurons equal to the size of the

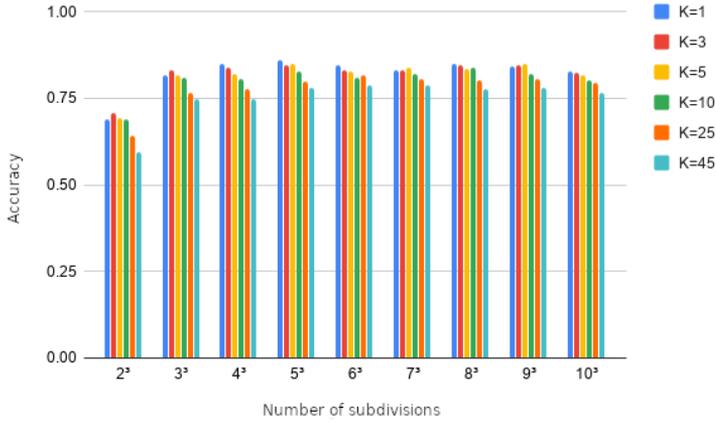


Figure 3.18: Comparison of ModelNet10 results for KNN with different k values and voxel subdivisions using euclidean distance.

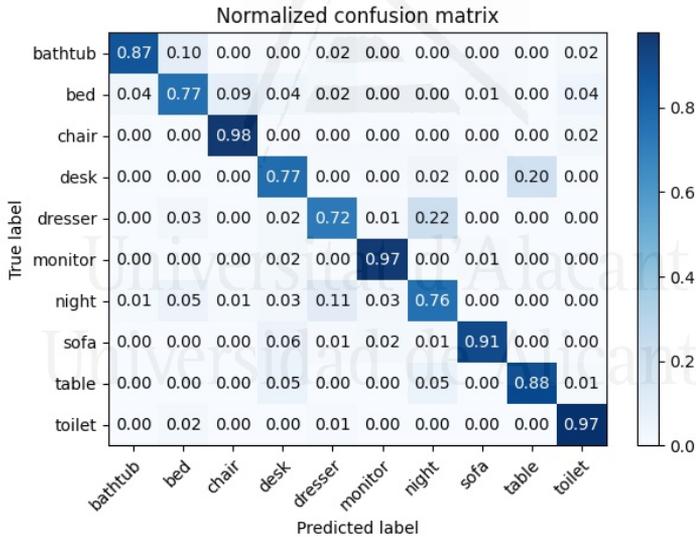


Figure 3.19: Confusion matrix for ModelNet10 using KNN with $k = 1$ and 5^3 subdivisions. 86,01% accuracy

feature vector (that depends on the voxel resolution) with sigmoid activation, followed by an output layer of 10 neurons (one for each category) with softmax activation. The best results were obtained with 9^3 subdivisions, which yielded an accuracy of **90,75%**. The corresponding confusion matrix is shown in Figure 3.20.

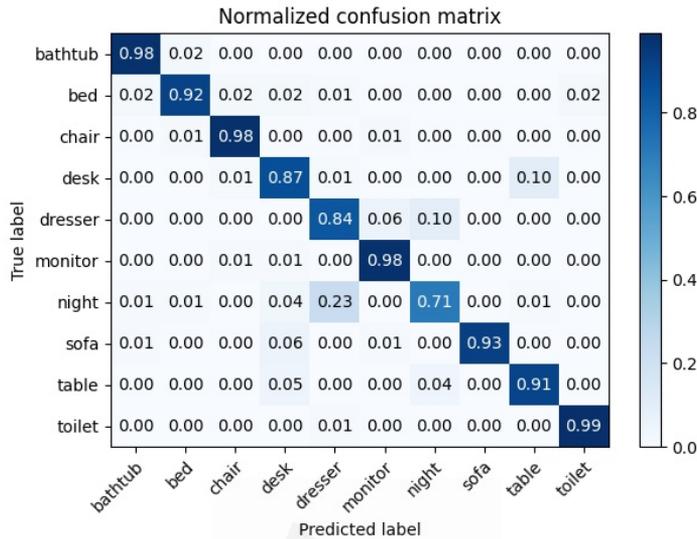


Figure 3.20: Confusion matrix for ModelNet10 with a fully-connected network and 9^3 voxel resolution. **90,75%** accuracy.

After observing the learning curves, we noticed a little overfitting, even though it is a very simple network, so we used regularization techniques such as Dropout [Srivastava et al., 2014]. This technique is very simple and consists of randomly disabling the connection between some neurons, so we force every neuron to learn something that contributes to the overall result of the network, preventing memorizing. After involving this technique, we achieved an accuracy of **90,97%**, but differences between train and test accuracy were significantly lowered, as depicted in Figures 3.21, 3.22, 3.23, 3.24.

The last classifier we tested was Support Vector Machines (SVM). With this method we achieved the highest accuracy, **92,84%**, with 5^3 subdivisions. In the light of the results, we can determine that SVM is the best classifier for our descriptor.

Figure 3.25 shows the confusion matrix regarding this experiment, where we can notice that a great part of the errors are misclassifications between conflicting classes, as exposed in Section 1.2.10.1. This fact shows that our system is working properly, as the confusions are between classes that are visually and semantically similar, which is very hard to tell apart

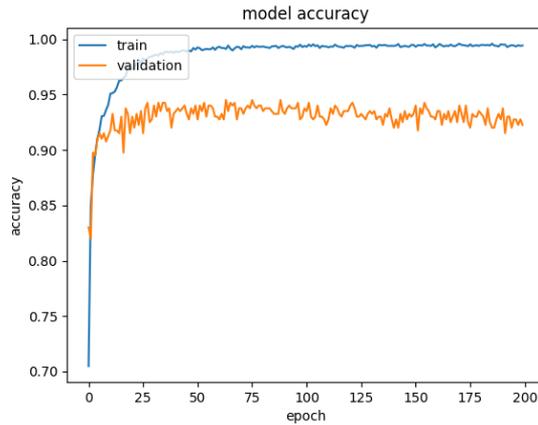


Figure 3.21: Accuracy before applying Dropout

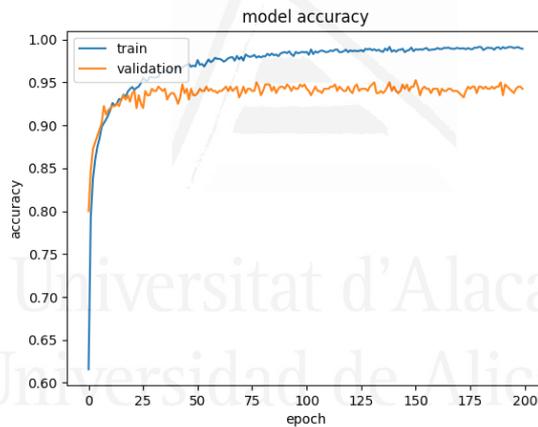


Figure 3.22: Accuracy after applying Dropout

even for humans.

3.3.2.3 Results on ModelNet40 using VFD

In view of the success of the SVM classifier for *ModelNet10*, we decided to move on to the extended version of the dataset that involves 40 different classes. We carried out different experiments with a range of voxel resolutions, as shown in Figure 3.26. Best results were obtained with 7^3 subdivisions, which confusion matrix is depicted in Figure 3.27, with a test

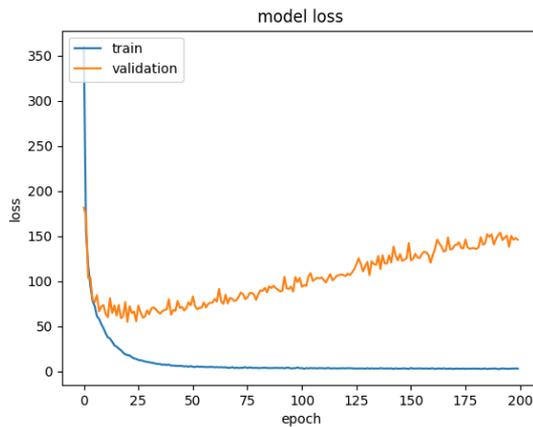


Figure 3.23: Loss before applying Dropout

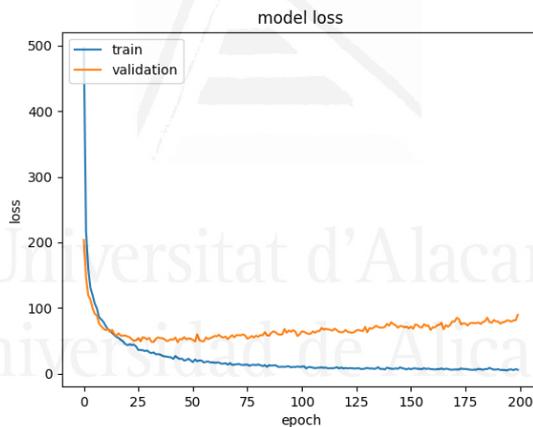


Figure 3.24: Loss after applying Dropout

accuracy of 88,74%.

In [Wu et al., 2019], we can find the reported accuracy of the main relevant papers for both ModelNet10 and ModelNet40. In Table 3.1 we offer a summary with the performance of some recent or relevant works for this topic. It is important to note that many of these methods rely on deep learning and multi-view representations, contrary to the proposed method that is a global descriptor based on fractal properties of the objects.

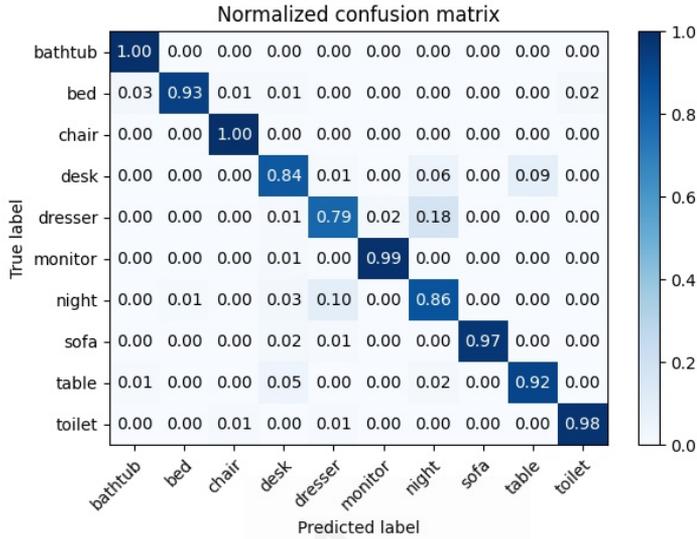


Figure 3.25: Confusion Matrix for ModelNet10 with SVM and 5^3 subdivisions. 92,84% accuracy.

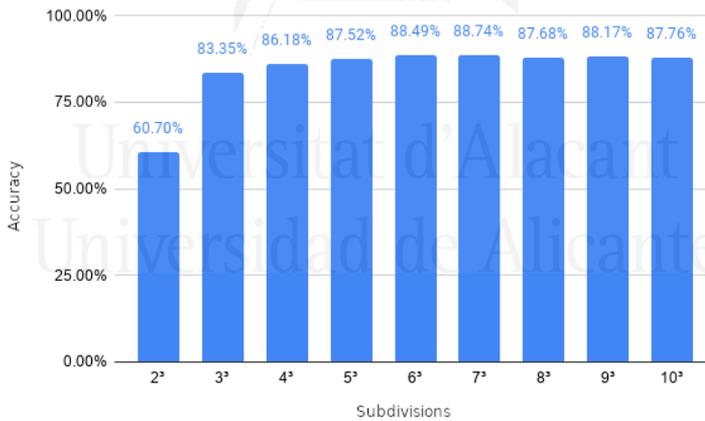


Figure 3.26: Accuracy for SVM classifier in ModelNet40 with different voxel resolutions.

3.3.3 Conclusions and future Work

The feasibility of the fractal dimension as global descriptor to recognize 3D objects has been demonstrated through the experiments carried out. The results obtained for the datasets *ModelNet10* and *ModelNet40*, used

Paper	MNet10 Acc	MNet40 Acc
RotationNet [Kanezaki et al., 2018]	98.46%	97.37%
SPNet [Yavartanoo et al., 2018]	97.25%	92.63%
SO-Net [Li et al., 2018a]	95.07%	93.40%
Point2Sequence [Liu et al., 2019a]	95.30%	92.60%
Lonchonet [Gomez-Donoso et al., 2017a]	94.37%	-
VFD	92.84%	88.74%
binVoxNetPlus [Ma et al., 2017]	92.32%	85.47%
Primitive-GAN [Khan et al., 2019]	92.20%	86.00%
VSL [Liu et al., 2018b]	91.00%	84.50%
OrthographicNet [Kasaei, 2019]	88.56%	-
3DShapeNets [Wu et al., 2015b]	83.50%	77.00%
PointNet [Garcia-Garcia et al., 2016]	77,6%	-

Table 3.1: Comparative of accuracy results on ModelNet benchmark

as benchmarks, have been successful, with a test accuracy of 92,84% and 88,74% respectively, which make the VFD comparable to the deep learning-based methods of the state of the art.

One main advantage of our method is that it is inherently agnostic to the to the density of points of the sample, number of points in the input cloud, sensor of choice, and noise up to a level because it is based on the fractal dimension of a set of points. This feature allows VDF to provide competitive accuracy on classification tasks. Another advantage is that our method is able to work on real life, sensor-provided point clouds as it do not relies on intermediate, rendered representations. It also does not require any powerful, specific hardware to create the descriptor and perform classification tasks on a reasonable amount of time, unlike the deep learning-based approaches. Nonetheless, our descriptor has linear complexity as the size of the descriptor is directly related to the resolution of the voxel grid clustering, as there is one component on the descriptor for each voxel. The resolution is also linked to the accuracy, as more complex objects would require more resolution in order to capture the finest details.

Throughout this section, different methods for calculating fractal dimension have been mentioned. It seems logical that if the box counting method has succeed for 3D object recognition, other methods could work too and even improve the results or efficiency, so we plan to test different

approaches as future work. In addition to that, we would explore method to limit the complexity of the descriptor in order to make it constant.

Another path to explore is to improve the implementation of the box counting method. We want to study the selection of a subset of points to approximate the line which slope is the fractal dimension, instead of applying least-mean squares minimization over all points. With this technique we could avoid the distortion produced by the leftmost points, as shown in Figure 3.28.



Universitat d'Alacant
Universidad de Alicante

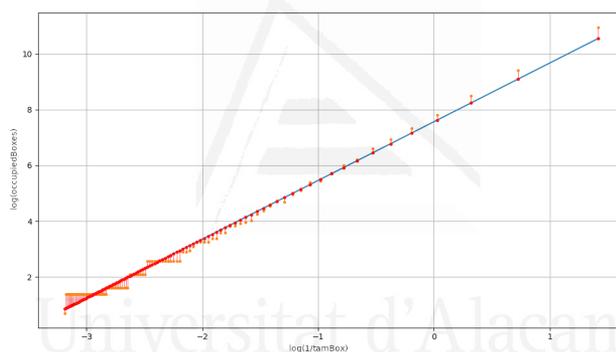


Figure 3.28: Errors with least-mean square minimization for box counting fractal dimension calculation.

Mapping of the environment

In this chapter, we describe the contributions made within the field of mapping of the environment. A brief introduction is provided in Section 4.1. Section 4.2 presents a system that exploits the representation capabilities provided by available pre-trained deep learning models to automatically label indoor environments and generate a 3D segmentation of the objects present in the scene. Section 4.3 proposes the integration of a domestic social robot into a Ambient Assisted Living (AAL) system, enriching the 3D mapping with the detection of elements and objects potentially dangerous to a person. Section 4.4 propose a mapping, navigation and planning system that makes use of a semantic location system to return the optimal route between rooms. Finally, Section 4.5 presents a solution to the problem of Pepper depth estimation by sensory fusion with monocular depth prediction.

4.1 Introduction

The use of appropriate environment representations is needed for most of the current robotic systems. Traditionally, environment representations have been limited to metrical maps that evolved from 2D to 3D with the release of affordable range sensors. In addition to this metric representation, semantic labels can be also used to represent rooms or scene categories. However, the location of relevant elements of the environment

should be explicitly provided, which involves human supervision and reduces the adaptability of the system to environment modifications.

The following sections propose different methods for mapping the environment that can be useful for a social robot in an indoor environment. On top of the location information of the metric maps, the localization of interesting and/or potentially dangerous objects in the environment is incorporated. In addition, a navigation system is proposed that makes use not only of the information from metric maps, but also allows planning using topological maps and the connectivity between rooms in a house. Finally, the last section includes a method to perform sensory fusion between different sources of depth images, taking advantage of the benefits of each one, which improves the mapping results by providing better quality point clouds.

4.2 3D object mapping using a labelling system

In this section, we propose to exploit the representation capabilities provided by available pre-trained deep learning models to automatically label indoor environments. We did not train our own CNN. Instead of that, we adopted the architecture defined by GoogleNet [Szegedy et al., 2015] as well as the pre-trained model they provide. This model was trained using the dataset ImageNet 2014 and it obtains a top-5 accuracy of 88.9%. It has to be emphasized that we chose this model over the rest because of its high accuracy rate and the fact that it provides object recognizing features for over 1,000 different classes, which will make our system adaptable and very context independent.

Our approach relies on the use of a mobile robot Pepper by Aldebaran Robotics. With its RGB-D sensor (Asus Xtion Pro Live) and its odometry system, it is suitable for performing a 3D registration of the environment.

Over this metric representation, we automatically integrate semantic labels that allow us to determine the most probable location of objects. This process successfully combines 2D semantic labeling with 3D registration in an automatic fashion. An overall scheme of the proposal can be seen in Figure 4.1.

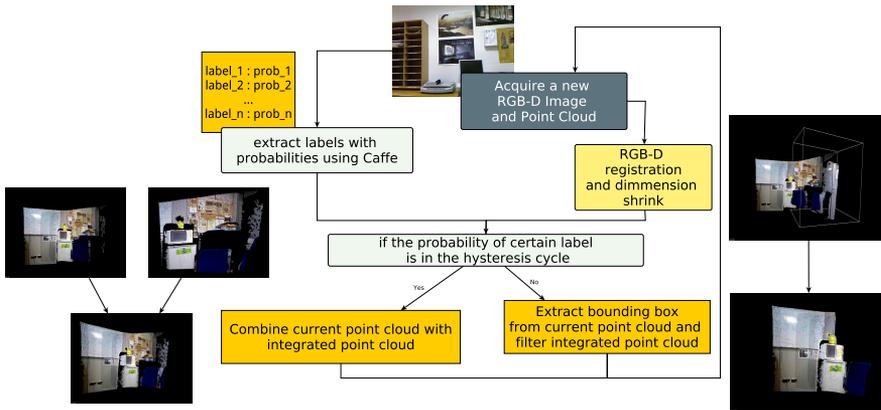


Figure 4.1: Overall pipeline of the proposal. When a new frame is captured, the RGB-D data is extracted and mapped to a point cloud, then the system registers it to a global coordinate frame. Alongside this process, it extracts the probability of finding objects using a CNN. If the probability exceeds certain threshold and fall into the hysteresis cycle (object is found in the scene), the current point cloud is merged with the point cloud that eventually will hold the 3D object. Otherwise, the bounding box of the current point cloud is calculated and used to crop the integrated point cloud.

4.2.1 Semantic labelling

Semantic labelling lets images to be described by means of a set of semantic concepts attributed to the scene perceived by the camera. This representation is suitable for human-robot interaction, as semantic terms can be easily included in human-robot interaction processes. The use of semantic labels also facilitates the understanding of robot surrounding, which may help to automatically determine the most appropriate robot behaviour in any scenario.

To implement this annotation process we make use of existing deep learning annotation tools. Deep learning techniques, and more specifically Convolutional Neural Networks (CNN [Lee et al., 2009]), allow the generation of discriminant models while discovering the proper image features in a totally unsupervised way, once the network architecture has been defined. This is possible nowadays thanks to the availability of huge image datasets annotated with large and miscellaneous set of semantic labels, which efficiently permits the training of these discriminative classification models.

In this section, we focus on the application of existing CNN models. The definition and building of these CNN models is beyond the scope of this chapter, so we refer the reader to [Bengio, 2009] for a more detailed view of deep learning in general and, to [Jia et al., 2014] for a better understanding of these CNN models.

4.2.2 3D mapping of the environment

For the generation of a 3D map of the environment, we have explored two alternatives: using Pepper's odometry data and using visual registration techniques. Finally, for the best results, we can use the last one as a refinement of the first one.

4.2.2.1 Registration by odometry

We can know the absolute position and orientation of the camera through the successive relative transformations from the odom frame, that indicates the transformation and orientation change of the robot's base from the initial moment, to the reference frame of the camera.

ROS interface uses the Unified Robot Description Format (URDF) as a *XML* representation of the kinematic, dynamic and sensorial model of the robot provided by Aldebaran for publishing the relative transformations between every Pepper's joint.

In order to get the global position and orientation of the camera, we must use the following relative transformations between frames:

1. **Odom -> Base_link.** Transformation from the reference frame of the first robot point known as *Torso*, placed at -38mm X and 169.9mm Z from head's Yaw axis origin, to the reference frame of the same point at the current time. This transformation is received from the odometry of the base.
2. **Base_link -> Torso.** Transformation from the reference frame of the point *Torso* and the center of its real torso.
3. **Torso -> Neck.** Transformation from the reference frame of its torso and its neck's one.

4. **Neck -> Head.** Transformation from the reference frame of its neck and its head's one.
5. **Head -> CameraDepth_frame.** Transformation between the reference frame of its head and its physical depth camera's one.
6. **CameraDepth_frame -> CameraDepth_optical_frame.** Transformation between the reference frame of its physical camera and the one used in camera's software for its data capture.

For the generation of the 3D map, we apply the calculated global transformation matrix to the current pointcloud and accumulate it in the building map. In Figure 4.2(a) a sample of this kind of registration is shown.

4.2.2.2 Visual registration

We have followed the pairwise registration pipeline described in Section 2.5.2 using PCL for 3D data processing, summarised in Algorithm 1.

Algorithm 1 Pairwise point cloud registration algorithm

Require: Dataset X .

Ensure: 3D Map M .

- 1: Total transformation $T_T = I$
 - 2: 3D Map $M = \emptyset$
 - 3: **for** $i = 1$ **to** $t - 1$ **do**
 - 4: Extract features $C_i = \{c_1^i, c_2^i, \dots, c_m^i\}$ from data x_i
 - 5: Extract features $C_{i+1} = \{c_1^{i+1}, c_2^{i+1}, \dots, c_m^{i+1}\}$ from data x_{i+1}
 - 6: Make pairings $P = \{p_1, p_2, \dots, p_t\}$ between C_i and C_{i+1} characteristics. Each matching is a pair of features from each dataset. $p_j = \{c_a^i, c_b^{i+1}\}$
 - 7: Obtain the best transformation T_i that explains the matches.
 - 8: Obtain total transformation $T_T = T_T * T_i$.
 - 9: Apply to the data set C_{i+1} the transformation T_T to place it in the coordinate system of C_1 . Accumulate the transformed data set into M .
 - 10: **end for**
 - 11: **return** M
-

In order to check that it works correctly, several pointclouds have been extracted from a house simulation in *Gazebo* and this method has been ap-

plied. Results shown in Figure 4.2(b) are satisfactory, so visual registration can be incorporated to our system.

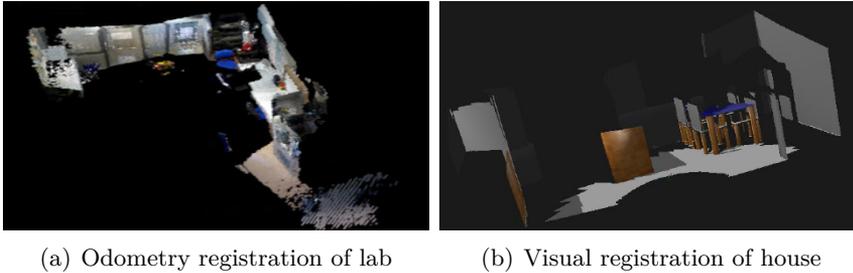


Figure 4.2: Examples of registration

4.2.3 Global image annotation using CNN

Let $Label = \{label_1, \dots, label_{|Label|}\}$ be the set of $|Label|$ predefined semantic labels, and I an input image. The direct application of the existing CNN models on I generates a descriptor $d_{CNN}(I) = ([prob(l_1), \dots, prob(l_{|L|})])$, where $prob(l_i)$ denotes the probability of describing the image I using the i -th label in $Label$. This obtains a representation similar to the Bag of Words (BoVW [Sivic and Zisserman, 2003, Martínez-Gómez et al., 2016]) approach.

We use the Deep Learning framework *Caffe* with a pretrained model of *GoogleNet* described in [Szegedy et al., 2015], that was top-5 in *ILSVRC 2014* challenge, consisting in image categorization between 1000 categories of image database *ImageNet* [Deng et al., 2009]. To check and improve the precision of this classifier, we have made a brief test with a sample image (in Figure 4.3) and applied different approaches to enhance classification results.

In the first column of Figure 4.4 we can see an example of *Caffe* classifier's execution on the sample image with the 10 higher scored tags. For better differentiation of higher probability tags, we propose to perform a normalization of their values like in Equation 4.1.



Figure 4.3: Sample image for classifier testing

$$prob_norm_i = \frac{prob_i}{\sum_{i=1}^n prob_i} \quad (4.1)$$

This way, we achieve a slight probability enhance of higher scored tags, as shown in the second column of Figure 4.4. If we analyze the output tags, we can observe that there are cases where very similar entities are separated in different classes and even represent the same physical object. This fact can turn into a issue because probability can be divided between these tags when their same referred object appears in the image.

To solve this, we propose the creation of a rename list that unifies the probability of these tags under a unique entity. In the third column of Figure 4.4 the results of this change can be seen. Due to the method and the examples used on the model training, it tends to value with a high probability the biggest and centralized object in the scene. As a consequence, the presence of another interesting objects could be ignored. For this reason, we propose a solution doing partial and weighted classification over multiples parts of the image, with variable-size windows. Finally, these classifications are fused in a final one.

In the last column of Figure 4.4 we can observe the classification result for this alternative. In this case, the probability of microwave has experimented a great growth.

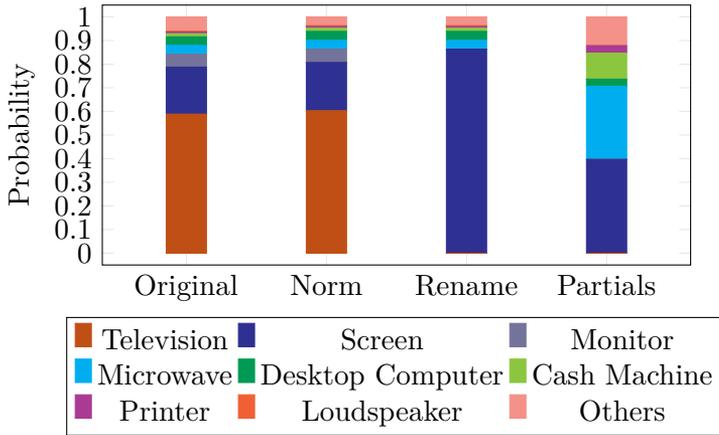


Figure 4.4: Comparative plot between classification alternatives

4.2.4 3D mapping with semantic labels

Given an image, the CNN model provides us with the semantic labels present in the image, so we can expect that the semantic label corresponds to an existing object in the image as well. Unfortunately, we are not provided with the position of the image, and therefore the environment, where the object may be placed. So, we propose to move the camera around (left, right, up and down) to look for the limit when the object disappears. Imagine that we move the camera to the left. In that situation, it is expected to have high probability value of a given label (if the object is present in the image). However, when the object disappears, the associated probability decays. If we find the limit when the object appears and disappears, we have a way to determine where the object is (and where it is not). We just need to accumulate all the point clouds where the associated probability to a given semantic label is above a threshold.

With the transformations calculated in the registration phase, we can know the direction of the camera and the location in the 3D scene of the RGB image, because it is aligned with depth image. This information is very important for the realization of this method.

When an object is imaged in the current scene the CNN returns a peak probability. As the object disappears of the scene as the camera moves, this probability descends gradually. To deal with this event we propose the

application of an hysteresis cycle. We apply the hysteresis cycle as follows: first we set a probability peak threshold Th_1 and a second lower threshold Th_2 . When a probability of a given label exceeds Th_1 , it means that this object is in the scene so this way it enters the hysteresis cycle. From this point we assume the next captures will have this object as well, and we accumulate them to the point cloud of this object. When the probability of this label is below a second much lower threshold Th_2 , it means that this object is no longer in the scene. This event sets the end of the hysteresis cycle so we stop accumulating new point clouds, and we use the last of them to eliminate the exceeding points that do not belong to the object itself but to the background or the rest of the scene.

For false activation or deactivation peaks filtering, we can provide a safety margin and force their repetition for a determined number of successive captures.

The task of removing the exceeding points is made with the *Conditional Removal* filter of *PCL*, that allows the removal of points that does not satisfy a certain conditions. If we calculate a *Bounding Box* over the pointcloud, we obtain a prism with the maximum and the minimum points of it. With these limit points, we can establish the proper conditions in order to remove the exceeding points in the accumulated map.

If there were two -or more- visible objects in the scene, they are going to be detected as two different object as long as they exceed the detection threshold, and they will be segmented and labeled accordingly. Note that the detection probabilities obtained by the CNN are fully distributed, if we have a lot of objects in the scene, it is possible that none of them meet the detection threshold, but this case requires a huge amount of objects in the scene. Also, if this case is expectable, the detection threshold must be set accordingly.

Despite of the improvements in object recognition explained in previous sections, it is still complicated to capture the exact moment where the object has disappeared from the image. In order to try to avoid its partial elimination from 3D reconstruction, we propose to reduce cutting area according to the movement flow of the camera, as indicated in Algorithm 2.

The explained above lets us to distinguish between objects of different

Algorithm 2 Algorithm for cutting points to be made on the scene

Require: T_{prev} : Absolute transformation matrix of the previous scene in time.
 T_{curr} : Absolute transformation matrix of the current scene. N : Number of frames that have elapsed between T_{prev} and T_{curr} C : Current point cloud, recorded with respect to the world. $coef$: Constant that determines the portion of the total length in each dimension of the point cloud N to decrease the cut-off.

Ensure: P_{min} : Minimum end of the cut prism. P_{max} : Maximum end of the cut prism.

- 1: Get PC_{min} and PC_{max} , minimum and maximum points of the point cloud C .
 - 2: Get PC_{mean} , midpoint between PC_{min} and PC_{max} .
 - 3: Get T_{rel} , relative transformation in the global reference frame applied to T_{prev} in order to achieve T_{curr} . $T_{rel} = T_{actual} * T_{prev}^{-1}$
 - 4: Get $trans_x$, $trans_y$ and $trans_z$, translation values for x,y,z of T_{rel} divided by N to obtain the mean translation speed.
 - 5: Get rot_r , rot_p and rot_y , roll, pitch and yaw rotation values of T_{rel} divided by N to obtain the average rotational velocity.
 - 6: Build T_{vel} , transformation matrix including the translations and rotations calculated in the previous steps.
 - 7: Get PV_{mean} by applying the transformation matrix T_{vel} on PC_{mean} .
 - 8: Get $score_x$, $score_y$, $score_z$ as the difference between PV_{mean} and PC_{mean} in each of their coordinates.
 - 9: Get $norm_x$, $norm_y$, $norm_z$ as the normalised scores resulting from dividing the absolute value of each of them by the sum of the absolute value of the three.
 - 10: Initialise the cut-off points $P_{min} = PC_{min}$ and $P_{max} = PC_{max}$
 - 11: **if** $score_x > 0$ **then**
 - 12: $P_{min}.x = P_{min}.x + \frac{norm_x * long_x}{coef}$
 - 13: **else**
 - 14: $P_{max}.x = P_{max}.x - \frac{norm_x * long_x}{coef}$
 - 15: **end if**
 - 16: **if** $score_y > 0$ **then**
 - 17: $P_{min}.y = P_{min}.y + \frac{norm_x * long_y}{coef}$
 - 18: **else**
 - 19: $P_{max}.y = P_{max}.y - \frac{norm_x * long_y}{coef}$
 - 20: **end if**
 - 21: **if** $score_z > 0$ **then**
 - 22: $P_{min}.z = P_{min}.z + \frac{norm_x * long_z}{coef}$
 - 23: **else**
 - 24: $P_{max}.z = P_{max}.z - \frac{norm_x * long_z}{coef}$
 - 25: **end if**
 - 26: **return** P_{min} and P_{max}
-

classes. However, if we want to discriminate between instances of the same class, we must follow a few more steps when the activation threshold has been exceeded. In order to select the specific object that is being observed, we must make use of *Bounding Boxes* and distances between centroids for instance selection, as explained in Algorithm 3.

Algorithm 3 Algorithm for selection between objects of the same class

Require: C : Last point cloud captured by the camera. V : Point cloud vector of the different objects of the same class recognised so far.

Ensure: $index$: Index of the object within the vector V_{obj} being detected.

In case none is found, the output will be -1 and a new object must be created in the vector.

```

1: Output initialisation:  $index = -1$ 
2: Initialisation of the minimum distance found:  $minDist = maxValue$ 
3: Get  $PC_{min}$  and  $PC_{max}$ , minimum and maximum points of the point cloud  $C$ .
4: Get  $PV_{vmin}$  and  $PV_{vmax}$ , vectors with the minimum and maximum points in the point cloud of each element of the  $V$ .
5: Get  $C_c$ , centroid of the scene's point cloud.
6: for  $i = 1$  to  $tam(V)$  do
7:   Get  $PO_{imin}$  and  $PO_{imax}$ , minimum and maximum points of the point cloud of the element  $i$  element of  $V$ .
8:   Check for X-axis intersection.  $I_x = max(PC_{min}.x, PO_{min}.x) <= min(PC_{max}.x, PO_{max}.x)$ 
9:   Check for Y-axis intersection.  $I_y = max(PC_{min}.y, PO_{min}.y) <= min(PC_{max}.y, PO_{max}.y)$ 
10:  Check for Z-axis intersection.  $I_z = max(PC_{min}.z, PO_{min}.z) <= min(PC_{max}.z, PO_{max}.z)$ 
11:  if  $I_x$  AND  $I_y$  AND  $I_z$  then
12:    Calculate the centroid of the object's point cloud  $C_o$ .
13:    Calculate  $dist$  Euclidean distance between  $C_c$  and  $C_o$ .
14:    if  $dist < minDist$  then
15:       $minDist = dist$ 
16:       $index = i$ 
17:    end if
18:  end if
19: end for
20: return  $index$ 

```

Fig. 4.1 shows a graphical scheme of the proposed method. The algo-

rithm explaining all the phases of the method is detailed in Algorithm 4. The use of semantic labels has a clear advantage against classical methods, like those using visual features to identify objects. CNN has the ability of recognizing the same category for two objects even when the visual appearance is completely different.

4.2.5 Experiments and results

Two different experiments have been made in order to evaluate the performance of the proposal method.

Most of the experiments were carried out with the robot Pepper using ROS. Our method uses the 3D information from the depth camera. However, a big issue with that camera has been detected: pointclouds generated from it are so much distorted, as explained in previous sections. This error, of unknown origin, is not only of our robot. The same issue has been noticed in different owners. It makes mapping tasks impracticable.

To solve this problem, an external Xtion has been placed on Pepper's head as a provisional solution. ROS with *OpenNI2* has been installed under *Raspbian* connected to laboratory's wifi for depth and colour images publishing.

Due to performance issues, pointcloud generation is made in the main PC from the depth and rgb images published by the camera using processing modules of OpenNI2. The activation process of the external camera through ssh and the pointcloud generator has been included in a ROS launcher, so that it is transparent for the consumer of the pointclouds.

4.2.5.1 Object recognition over robotic platform

In this case, we use robot odometry as registration system for the scene shown in Figure 4.5. Due to the problem explained in previous sections, we use an external *Xtion* over Pepper's head, so there might be some inaccuracies because of slight camera deviations as a result of continuous movement of the head. Camera movement has been done centred on microwave.

In Figure 4.6, it appears the incremental registration of the scene. Some degree of overlapping is seen in microwave's area, but its result is quite good

Algorithm 4 Algorithm to build the 3D map from semantic labels

Require: Th_1, Th_2 : set of activation and deactivation of hysteresis cycle, one for each label. $\{PC_{label_i} = \emptyset\}$ set of pointclouds, one for each label. $\{hyst_{label_i} = false\}$ set of boolean variables, one for each label, that represents hysteresis cycle activation. $\{idetec_{label_i} = MAX_N_DET\}$ set of boolean variables, one for each label, that represents the remaining detection tries until cycle activation. $\{icut_{label_i} = MAX_N_CUT\}$ set of boolean variables, one for each label, that represents the remaining cutting tries until cycle deactivation.

- 1: **loop**
- 2: Get PC_j , registered 3D pointcloud and its associated RGB image.
- 3: Get $LabelSet = \{(label_k, prob(label_k))\}$, set of labels with their associated probabilities from classifier for RGB image.
- 4: **for** every label $label_m$ **in** $LabelSet$ **do**
- 5: **if** $prob(label_m) > Th_1$ and $!hyst_{label_m}$ **then**
- 6: **if** $idetec_{label_i} > 0$ **then**
- 7: $idetec_{label_i} = idetec_{label_i} - 1$.
- 8: **else**
- 9: Add the current pointcloud to the corresponding object map:
 $PC_{label_m} += PC_j$.
- 10: $hyst_{label_m} = true$.
- 11: $idetec_{label_i} = MAX_N_DET$.
- 12: **end if**
- 13: **else if** $prob(label_m) > Th_2$ **then**
- 14: **if** $hyst_{label_m}$ **then**
- 15: $PC_{label_m} += PC_j$.
- 16: **else**
- 17: $idetec_{label_i} = MAX_N_DET$.
- 18: $icut_{label_i} = MAX_N_CUT$.
- 19: **end if**
- 20: **else**
- 21: **if** $hyst_{label_m}$ **then**
- 22: **if** $icut_{label_i} > 0$ **then**
- 23: $icut_{label_i} = icut_{label_i} - 1$.
- 24: **else**
- 25: Get corrected Bounding Box (BB) of PC_j .
- 26: Remove points of PC_{label_m} inside BB.
- 27: $hyst_{label_m} = false$.
- 28: $icut_{label_i} = MAX_N_CUT$.
- 29: **end if**
- 30: **else**
- 31: $idetec_{label_i} = MAX_N_DET$.
- 32: **end if**
- 33: **end if**
- 34: **end for**
- 35: **end loop**
- 36: **return** PC

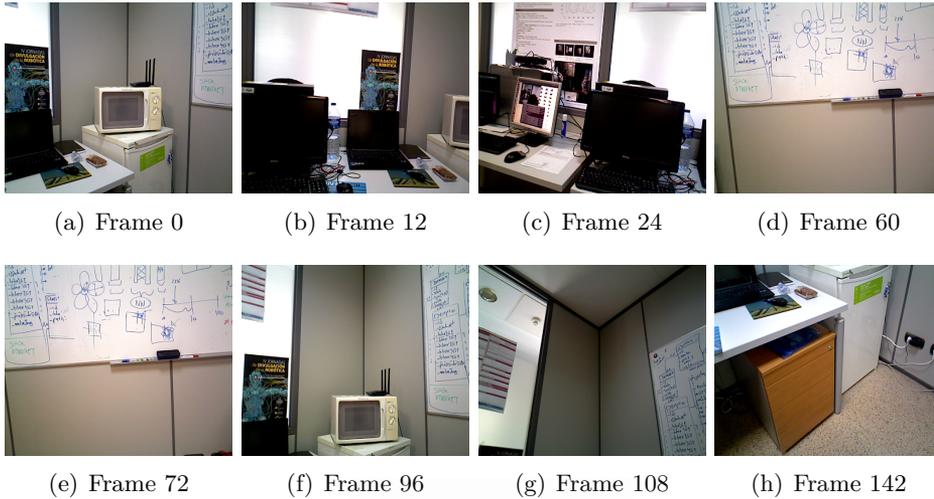


Figure 4.5: RGB images of different moments of the experiment

for our purposes.

Afterwards, Figure 4.7 shows the recognition process for microwave. One of the most critical moments happens at the first cutting stage on left movement. Our recognition system stops detecting microwave so early, when there is an important part of the object in the image, so the cutting process removes a small part of it. We can see some imperfections in the generated cloud, due to our less aggressive cutting policy. However, it could be completely isolated with clustering techniques.

In Figure 4.8, the probability evolution of microwave detection in the scene is shown. Again, we can see detection peaks and big falls corresponding to the four cutting moments when the object disappears.

4.2.5.2 Recognition of multiple object instances of the same class

In this case, this experiments tries to prove the recognition of multiples instances of the same object. It has been made with a *Xtion* over Pepper's head. The robot has made a camera moment centred on the first screen, as shown in Figure 4.9.

In Figure 4.10, it appears the incremental registration of the scene.

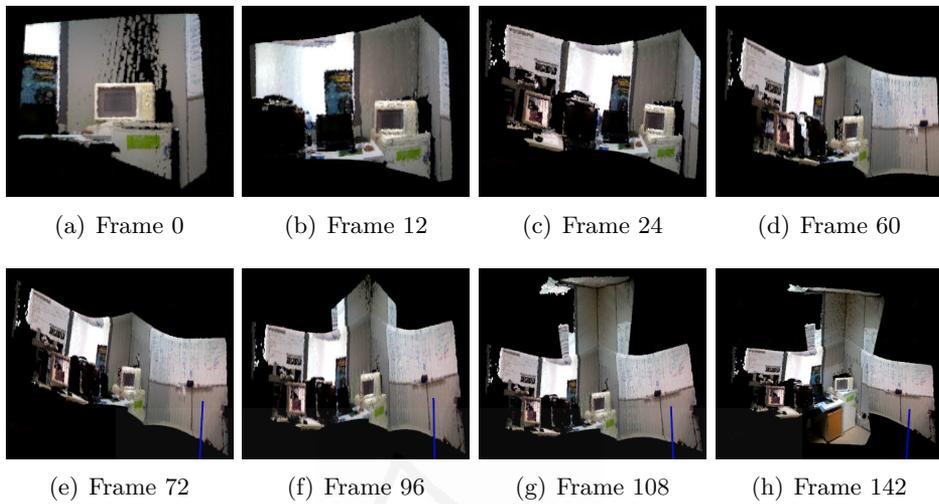


Figure 4.6: Incremental registration of the scene of the object-on-robot recognition experiment

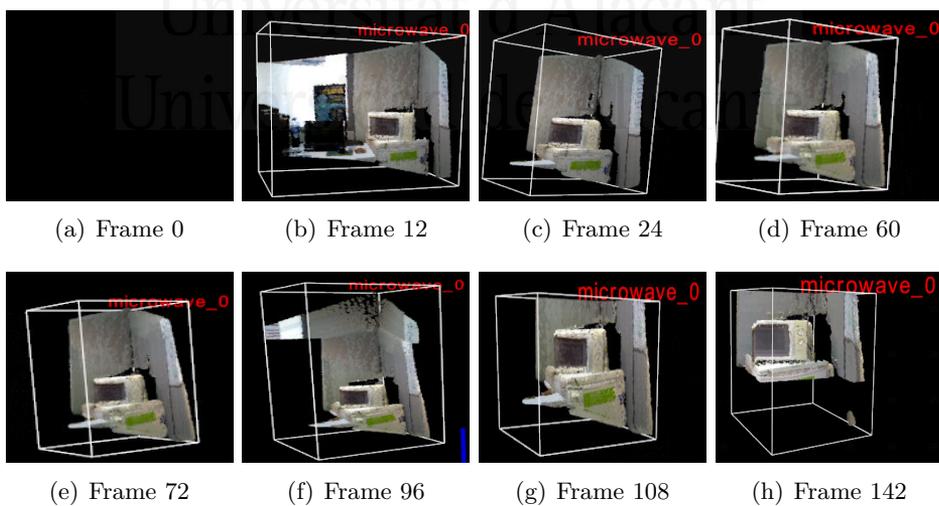


Figure 4.7: Objects recognised in the object-on-robot recognition experiment

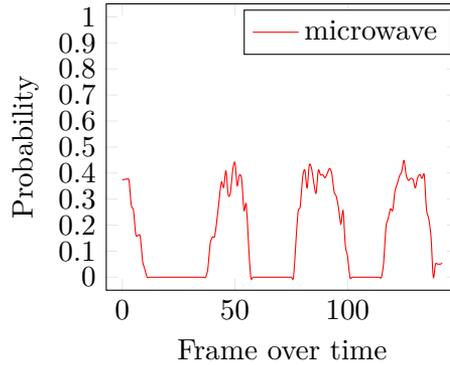


Figure 4.8: Probability log for the microwave detection

Some degree of overlapping is seen as commented in Section 4.2.5.1.

Then, Figure 4.11 shows the recognition process for screens. We can see the moment when another screen instance is added, because the first one is so far from the new detection. After the recognition of the second one, camera movement go back and focus the first one, and our system identifies it correctly. Due to the nature of the capture, the first screen is better delimited, because it has experimented cuts in 4 directions, and not only one as the second one.

Figure 4.12 shows the probability evolution of screen detection in the scene. It can be noticed that our system recognises fairly better the first screen - as seen in its higher probability peaks - because it is more isolated and has appeared several times in the middle of the scene. The second one has been observed in more difficult circumstances, only in the right corner of the image.

4.2.6 Conclusion

Starting from the primitive system with preset turns over Z axis, published in [Rodriguez et al., 2016] and revised in [Escalona et al., 2017c], in this chapter we have extended and generalised it for 6 DOF. We do not have preset movements any more, so we have implemented 2 techniques for cloud registration: visual and odometer registration. Due to these changes, we have been able to incorporate our system on board a mobile robot.

The system is able to build with success a 3D object map using an-

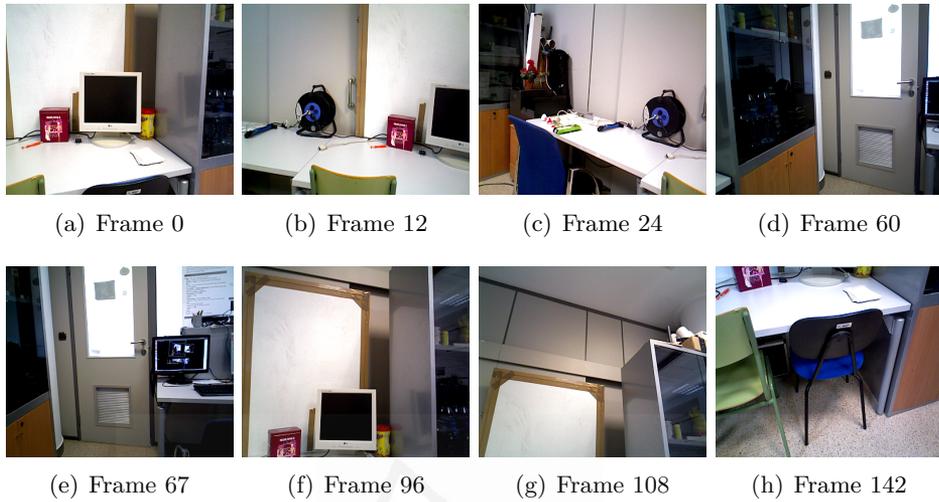


Figure 4.9: RGB images of the scene of the experiment on the recognition of multiple instances of objects of the same class

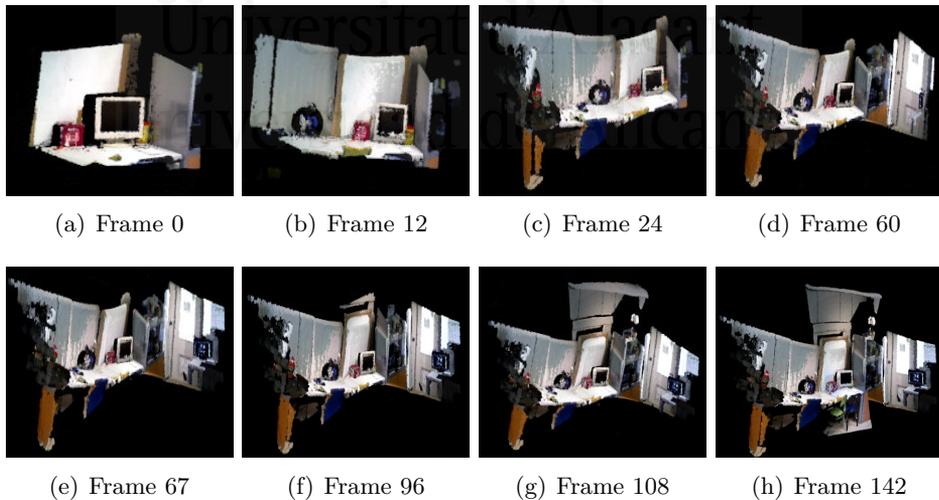


Figure 4.10: Incremental registration of the scene of the experiment on the recognition of multiple instances of objects of the same class

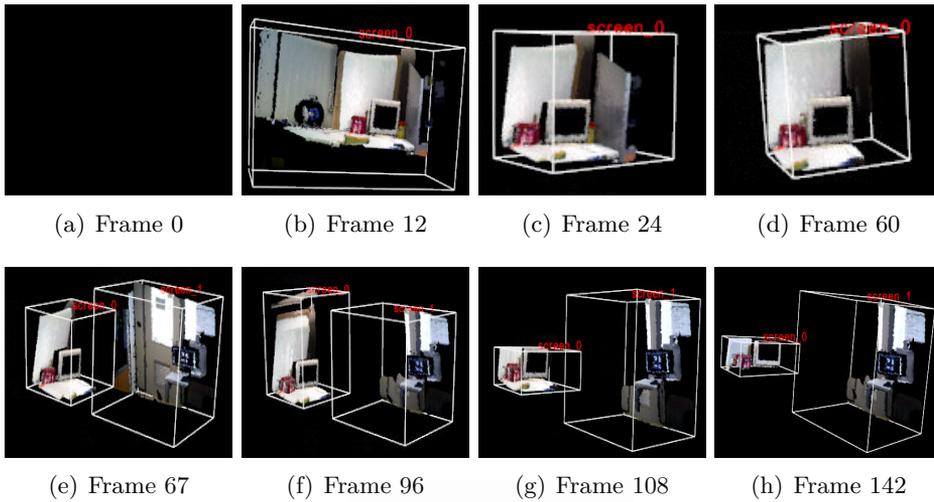


Figure 4.11: Objects recognised in the experiment on recognition of multiple instances of objects of the same class

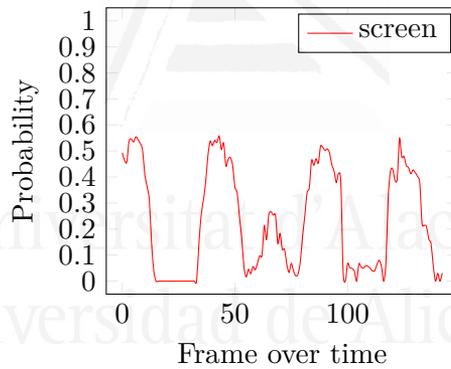


Figure 4.12: Probability log for the experiment on the recognition of multiple instances of objects of the same class

notations from a predefined CNN model. In future works, we will study different CNN architectures, 3D object recognition techniques and dynamic mapping.

4.3 Enhanced Ambient Assisted Living with a social robot

AAL provides innovative approaches to the challenges of an aging population and physically or mentally challenged individuals, helping them to stay active longer, remain socially connected, and live independently into old age.

To do this, a number of cameras are usually placed on the ceiling of rooms in order to cover as broad an area possible using the minimum number of devices. An external system takes camera feed and is able to detect potentially harmful places, a person falling or extended stays in a specific room, among other events. Nonetheless, smaller sources of danger such as electric panels, sockets, around radiators or the oven remain unnoticed due to the placing of the cameras and the relatively small size of the mentioned elements. These zones are currently introduced manually into the system. Although it is not ideal, this is acceptable due to the fixed nature of the sources of danger, which will not change position over time.

However, there exist common threats like knives, a dog, or a robot vacuum cleaner which, in addition to their small size, are also non static, i.e., their position could change over time. In these cases, it is not viable to manually set fixed zones of danger. Finally, it is worth noting that there will likely exist occluded areas caused by persons or furniture or even the field of view of the camera. In this case, it is impossible to detect any sources of danger.

The main contribution of this section is the integration of a domestic social robot into the AAL system to detect the dangerous elements that cannot be detected by cameras alone due to their size, itinerant nature or being located in an area that is occluded to the fixed cameras.

4.3.1 System description

The proposed system aims to integrate a domestic robot into an AAL environment. As mentioned earlier, one of the main tasks of AAL is to detect potentially dangerous elements. However, there are a number of

threats that will be unnoticed by the cameras of the system due to their relatively small size, or because they are in an occluded zone or are itinerant. In order to enhance the AAL system by making it aware of these threats, we propose the use of a mobile robot to detect potentially dangerous areas that the fixed cameras are not able to sense, and, thus, improve the performance and robustness of the system.

There are two main elements in this approach. The first is the AAL environment which consists of a number of cameras fixed to the ceiling that are able to precisely localize the persons in the scene, and to issue alerts when the person is in danger. The second is a robot that is in charge of continuously discovering new threats in the environment, and sending them to the AAL system in order to incorporate these threats into its alert system. In the following subsections, both subsystems are described in detail.

The mobile robot of choice is Pepper. Pepper is a social robot manufactured by Softbank Robotics. It features a light-duty onboard computer which is able to perform simple tasks. It is able to move in planes, like the floor of a house, and also integrates color and depth cameras. We used the Pepper robot to implement our system, but any robot with color cameras, a mobile base and a depth camera or laser could be used.

The AAL system was developed by *Pentalo Labs* company, so we will focus on the systems implemented in the Pepper robot.

4.3.2 Using a robot for dangerous areas detection

There are a variety of situations in which the AAL may not detect dangerous objects and events. We propose the integration of a mobile robot into the AAL system that can detect these cases and send them to the AAL in order to improve its performance.

The mobile robot of choice must be equipped with color and depth cameras, and it is assumed that it can move in the environment. The robot runs the pipeline depicted in Figure 4.13, which is described in detail in the following subsections.

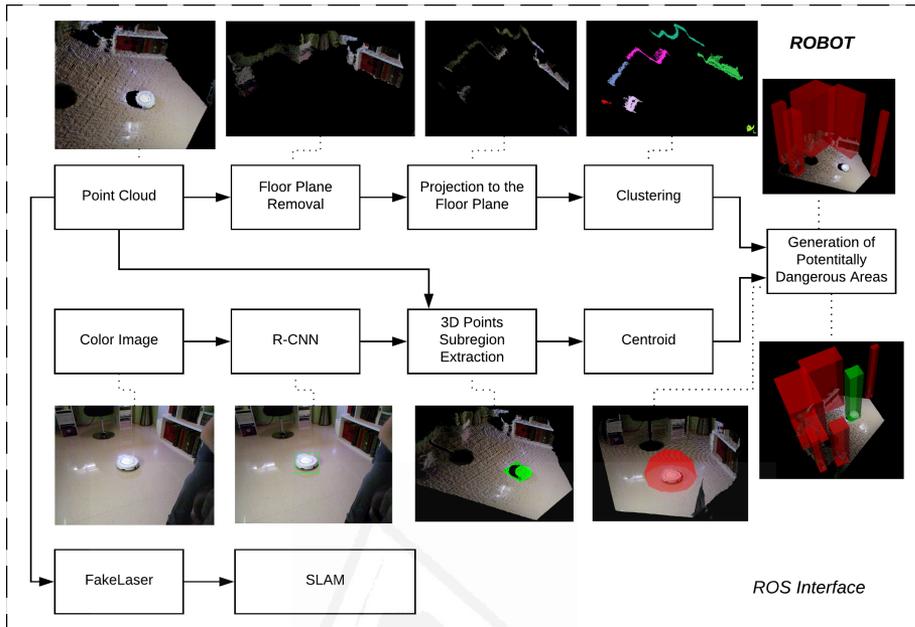


Figure 4.13: Description of the OGT and SOD algorithms pipelines which run in the robot.

4.3.2.1 Detection of objects on the ground

It is worth noting that all the objects above ground level could be a source of dangerous situations so must be considered by the AAL system. As explained earlier, the AAL system uses a static map of the environment which allows it to detect fixed obstacles like walls and doors. Nonetheless, moving obstacles are also a source of danger so we propose the following algorithm to detect fixed and moving objects and obstacles above the ground level. This algorithm runs in the mobile robot.

First, the robot captures a color image and the corresponding depth map. Using both data streams a color point cloud is generated. Then, the resultant point cloud of the scene is transformed to the global robot coordinate frame T^* ; $T^* = [R^*|t^*]_{4 \times 4}$. At the start of the algorithm, if it is the first frame, the transformation is the identity $T^* = I$, namely, the initial robot coordinate frame is assumed as the global robot coordinate frame. If it is not the first frame, the current transformation T ; $T = [R|t]_{4 \times 4}$ is

accumulated to the global transformation $T^* = T^* \times T$. To compute the current transformation T we used robot odometry. In this way, each point cloud $Pc = \{P_i\}; P_i = (x, y, x, r, g, b)$ is transformed to the robot global coordinate frame $Pc^* = Pc \times T^*$, thus creating a tridimensional map of the environment $M = M + Pc^*$.

The next step is to detect the floor plane, so we used RANSAC [Fischler and Bolles, 1981] to carry out this process. RANSAC is a model-fitting algorithm that takes a set of data and tries to fit it in a model. In our case, the input data is the recently acquired point cloud Pc^* and the model is a plane. As a result, this step returns the coefficients a, b, c, d of all the detected planes in the scene. Each plane is modeled as $ax + by + cz = d$. A simple test, which consists of checking whether the z component is about 0 within a threshold, is carried out to reject the planes that are not at ground level. As the planes are estimated over data gathered by the sensor, it is likely that the z component is not exactly 0, but very close to it. The points that lay in that approximate plane of the floor within a threshold are deleted from the point cloud. At this point, only the objects above ground level remain in the point cloud. This threshold prevents noise and possible artificial artifacts from being detected as obstacles. Then, the points of each object are projected to the previously computed ground plane. As a result, a 2D map of the obstacles above ground is obtained.

In the next step, a Euclidean clustering process is performed in order to segment and isolate the obstacles. The Euclidean clustering method essentially groups points that are close together. A cluster tolerance threshold is set so the points within this threshold are considered to be part of the same cluster. As a result, this process will return a cluster for each obstacle. A rectangle is then fitted to the points of each cluster. The rectangles are used to build the dangerous zones by extruding the potentially dangerous areas in the Z axis in order to transform the map to the tridimensional space once again. Note that the clusters are not intended to have a semantic meaning, we pursue the best possible geometrical fitting. Consequently, each cluster may not represent a specific object.

Finally, current dangerous areas are fused with previously detected dangerous areas if they overlap. This process is looped as the robot moves,

thus building a map of the environment and keeping track of the potentially dangerous areas. We named this algorithm the Obstacles over the Ground Tracker (OGT).

4.3.2.2 Superficial object detector

Although the OGT algorithm performs reasonably well for large obstacles, some dangerous areas will still not be detected, as stated before. For instance, objects like wall or floor sockets, electrical panels or vacuum cleaners are not detected by the described algorithm because they are too small to be sensed by the 3D camera. Hence, we propose the following pipeline to detect these cases. This algorithm is executed alongside the OGT.

First, the color image captured by the camera of the robot is fed to a Region-based Convolutional Neural Network (RCNN). The RCNN is able to return the bounding box and the category of the objects it detects. Then, the points inside each area of interest of the detected objects are extracted. This process is straightforward as the color image and the point cloud are registered beforehand. Next, for each subset of points inside the bounding boxes, the median-centroid is computed. This is done due to the presence of non-object points in the bounding boxes. As the bounding boxes are rectangles, the majority of the points belong to the object of interest, but there are still background points. A cube is then fitted for the points of each object but, this time, keeping the center of the cube in the previously computed median-centroid.

In this way, we can use the color information to detect these risks that are not sensed by the tridimensional sensor or ignored by the OGT, and build a more comprehensive map of the potentially dangerous areas. We named this algorithm the Superficial Object Detector (SOD).

For each subset of points inside the bounding boxes, the centroid is computed. Then a spherical potentially dangerous area is set which center is the centroid previously extracted. The radius of the spherical potentially dangerous area is computed as the maximum value of the difference of the maximum and minimum point in each dimension divided by 2 and multiplied by a scale factor. The scale factor controls how much surrounding of

the object is included in the potentially dangerous area.

Finally, the potentially dangerous areas detected by the OGT and OGT pipelines running in the robot are merged and sent to the AAL system in order to detect whether the person enters one of these zones.

It is worth noting that there are likely to be potentially dangerous zones that could be detected by different systems at the same time. For instance, walls are detected by the three methods. In addition, objects located on top of other objects are also redundantly detected. For instance, electric sockets in the walls or a knife on a table. The reason for not filtering these cases is twofold. First, the redundant detections improve the robustness of the system, and second, it could enhance the alerts emitted by the AAL system by adding semantic data.

4.3.3 3D world references registration between AAL and the robot

As mentioned before, the AAL and the Robot are continuously sharing information. For instance, the potentially dangerous objects detected by the robot are sent to the AAL. The robot transmits the tridimensional position of the objects in its own reference frame. Nonetheless, the event of a user trespassing the area near that object is detected by the AAL. Given this pipeline, this process can be only carried out if both systems are working in the same reference frame. This process is depicted in Figure 4.14.

It is worth noting that integrating two systems working on their own references and coordinates is not an easy task. In order to interact between systems, both need to be in the same system of coordinates with the same reference axis.

To solve this issue, both systems have been calibrated using the same coordinate reference. On one hand, the AAL detects a known pattern (both location and size are known, Figure 4.15(a)) and estimates its position using a Perspective-N-Point algorithm [Li et al., 2012]. This way, the camera is located within the coordinate frame of the pattern. The robot follows the same procedure. It solves the Perspective-N-Point problem to locate itself in the coordinate frame stated by the pattern (Figure 4.15(b)).

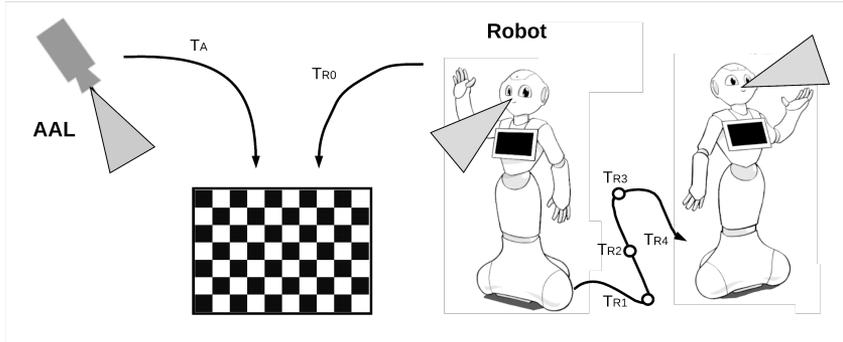


Figure 4.14: Diagram describing the method to compute the transformations between the AAL and the Robot that enables the correct sharing of the tridimensional position of the detected potentially dangerous areas.

This calibration procedure enables both devices (AAL and Robot) to share the same coordinate frame reference using the obtained transformation matrices T_A and T_{R0} .

This calibration step is performed once at the setup stage. However, if the robot moves, the transformation T_{R0} we previously computed is no longer valid. To solve this issue we rely on SLAM methods.

The localization of the robot within the environment is carried out using SLAM algorithms. Specifically, it uses the GMapping ROS Package [Huletski and Kartashov, 2016], which implements the Monte Carlo Localization algorithm. This algorithm uses laser scans as input. Nonetheless, the laser sensor of our robot is quite limited and noisy, we used the depth camera to simulate it. First, a depth map is captured using the aforementioned depth camera. Then, we extract the central row of the depth map. As the values of the depth map are in fact distances to the objects in the scene, the reinterpretation to laser scans is straightforward. This process is named “FakeLaser” in the Figure 4.13. As a result, this method provides additional transformations $T_{R1} \dots T_{Rn}$. These chain of transformations describe the position of the robot so they are used to compute the transformation between the coordinate systems of the AAL and the Robot even if it moves.

Summarizing, the AAL and the robot are both calibrated using a common pattern. As a result, both devices are localized in the same 3D coor-

dinate frame through the transformation matrices T_A and T_{R0} . This step allows the transformation of the 3D objects detected by the robot to the AAL coordinate frame. If the robot moves, the transformation is no longer valid, so we rely in the mentioned SLAM algorithm to compute additional transformations $T_{R1}...T_{Rn}$. The chain of transformations enables the transformation between the coordinate frame of the robot and the AAL even if the robot moves.



(a)



(b)

Figure 4.15: In order to integrate the potentially dangerous areas detected by the robot with the AAL system, both coordinate frames must be registered. We use a common pattern to calibrate both systems in the same coordinate frame.

4.3.4 Experimentation

In this section, we describe the experimentation of the OGT and SOD algorithms separately in order to validate the detection of potentially dan-

gerous areas.

It is worth noting that we used a Pepper Robot as the mobile robot of choice. This robot is equipped with a color and a depth camera, and is able to move and compute the transformation between two frames through self-odometry. Due to the limited computational power of the on-board processor, all the computation is executed on an auxiliary computer equipped with an Intel i5-3570 CPU, 16 GB DDR3 RAM and a NVidia 1080Ti GPU. The R-CNN implementation leverages the GPU for accelerated algorithms. Communication between devices is provided by ROS Kinetic [Quigley et al., 2009a], JdeRobot [Chakraborty and Canas, 2016] and Internet Communications Engine (ICE) [ZeroC, 2018]. The operating system of choice is Ubuntu 14.04.

To make the verification of the system as accurate as possible, a log system was developed. This system is able to record information from a set of devices and save all the information to the hard disk of the computer. Subsequently, the data can be replayed identically to how it was provided by the physical device. This procedure was applied in order to verify the precision of the situation detected in all the following experiments. Following this procedure, we can synthetically reproduce, real daily life experiments including external perturbation to the data to verify the robustness of the algorithm. An automatic evaluator has been created to ensure the precision of the experiment. This evaluator will reproduce the recorded log (which contains certain risky situation) 3000 times with different kind of noise. If the expected risky situation has been detected the test will be labelled as a success. Each test will require 3000 times the duration of the recorded log.

4.3.4.1 Objects on the ground tracker algorithm experiments

This algorithm takes the point cloud provided by the Pepper Robot as input. The point clouds are dense with a 640×480 resolution, meaning there are 307200 points in each. The first step of the OGT algorithm is the removal of the floor plane, which is performed with RANSAC. The target model is a plane and the inlier threshold is 3mm. The projection to the floor plane is straightforward and has no parameters. The Euclidean

clustering process rejects clusters with a of below 200 points. This is done to filter smallish clusters that could emerge due to the noise present in the point cloud. In addition, cluster tolerance is 3 mm, which is the distance between two adjacent points that are in the same plane. All these parameters were set empirically.

The robot was deployed in an office environment. It was placed in the center of the room and performed a 360 degrees turn by rotating its base. The OGT algorithm was executed for each frame.

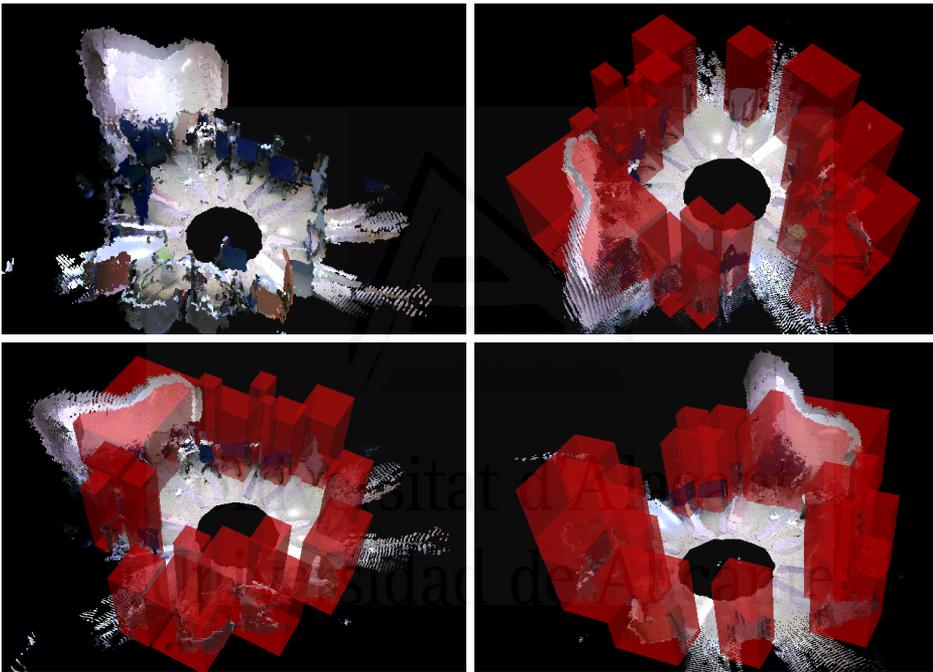


Figure 4.16: Results of detection of the potentially dangerous area in an office environment. The first image depicts the generated map of the environment. The second, third and fourth images show the results of the OGT algorithm from different points of view.

The results of this experiment are shown in Figure 4.16. As expected, the majority of the objects and obstacles were properly detected and the potentially dangerous areas created in the tridimensional map. Only three objects were not detected: a laptop power adapter, a knife and a bottle. They were ignored by the OGT algorithm due to the floor plane tolerance threshold. The bottle was also placed on the floor but its tridimensional

information was removed due to its minimum cluster size. It is worth noting that, due to the materials, the only part of the bottle sensed by the camera is the label. The plastic shape cannot be detected by the RGB-D sensor of the robot.

4.3.4.2 Superficial object detector algorithm experiments in office environment

As mentioned earlier, the color images and the point clouds are provided by the Pepper Robot at 640×480 resolution. The image and the point cloud are registered so, for each pixel in the color image, there is a corresponding point in the point cloud. Then, the color image is resized to a 416×416 , which is the input size of the YOLOv3 [Redmon and Farhadi, 2018] architecture. YOLOv3-416 is the chosen R-CNN implementation. This version provides 35 fps on a NVidia 1080Ti, which is suitable for real time uses, while currently being one of the most accurate architectures. The output of this architecture is composed of the bounding boxes of the detected objects and their corresponding category.

We trained our own model in order to detect the following objects: electrical panels, tangles of wires, wall and floor mounted sockets, knives, ovens, shoes, bottles, hobs, cats, dogs and robot vacuum cleaners. These categories include dangerous items and are common in houses and offices, so they are suitable for the experiments. Nonetheless, the model can be easily extended with new categories. The images needed to train the model were downloaded from free stock image sites and manually labelled. The dataset comprises about 11000 images, equally distributed across the categories. For training, 60% of the samples were used, 20% were used for validation and the rest for testing.

Instead of taking an already trained model, we trained one from scratch. As mentioned before, the system must detect the dangerous items that are likely to appear in indoor environments. The categories of objects we selected are: electrical panels, tangles of wires, wall and floor mounted sockets, knives, ovens, shoes, bottles, hobs, cats, dogs and robot vacuum cleaners. In total, we consider 11 categories. In order to train the model we build a custom dataset. To do so, 11,000 images were downloaded

automatically from free stock images websites. The number of images per category is balanced, so the dataset comprehends 1,000 images per class. The images were automatically downloaded in bulk by searching for the keywords mentioned before. A team of 5 human agents curated the dataset by ensuring that the images correctly depicted the categories and manually labelled the bounding boxes of the objects. Finally, we used a 60% of the samples for training, 20% were used for validation and the rest for testing. Once we built the dataset, we trained the architecture using the YOLO loss for 25,000 epochs. However, the best intersection over union score was reached at epoch 21,700. This model was selected and involved in the experiments. As a result, the system is able to accurately detect the considered dangerous items, and state the area of the image that depict them using an image of the scene capture by the Pepper robot.

The tridimensional sub-region extraction and the centroid computation have no parameters.

We tested the OGT in the same office environment in which we deployed the OGT system. We focused the detection on those that were not considered by the OGT system. We tested the following scenarios:

- Objects lying on the floor that are not considered by the OGT.
- Objects sitting on other objects and obstacles.
- Objects integrated into other objects and obstacles.

As mentioned, the OGT failed to detect some objects such as the laptop power adapter and a bottle on the floor. These objects remained unnoticed because certain steps of the OGT algorithms filtered them, but the OGT is able to properly detect them again.

In the OGT experiments, the knife and the power sockets were not detected alone but as part of the table and walls. In this case, the OGT does not contribute with new potentially dangerous areas but include semantic information on them. However, the semantic information of the object provides highly valuable data to enhance the alerts of the AAL system. For instance, if a falling event is detected by the AAL after the violation

of potentially dangerous area of the power socket, the patient may have suffered an electric shock.

The bottle was also ignored by the OGT. The depth sensor cannot compute the distance of plastic surfaces. The only part of the bottle represented in the point clouds is the label. Nonetheless, the amount of points in the label does not exceed the minimum point size of the clustering process of the OGT, so they are filtered. However, the color information correctly depicts the bottle, so the OGT system is able to detect it. Figure 4.17 shows these experiments.



Figure 4.17: Results of detection of the potentially dangerous area on an office environment. The first row depicts color images as captured by the mobile robot. The second row shows the potentially dangerous areas as returned by the OGT algorithm. Spheres are drawn for visualization purposes.

4.3.4.3 Superficial object detector algorithm experiments in home environment

Homes also have multiple sources of potentially dangerous areas such as ovens or electrical panels. Hence, in this experiment, we deployed the robot in an actual home environment and ran the OGT algorithm. We will focus on the same goals as in the last experiment:

- Objects lying on the floor that are not considered by the OGT.
- Objects sitting on other objects and obstacles.
- Objects integrated into other objects and obstacles.

In a home, there are not likely to be objects lying on the ground that would be a source of danger. For instance, the OGT algorithm only detected the family’s dog. Nonetheless, our algorithm also found a variety of potentially dangerous objects that are integrated into, or sitting on, other objects. For instance, the robot vacuum cleaner, the oven, the hob, electrical panels and electrical sockets, and knives were correctly detected as potentially dangerous areas. As mentioned earlier, some of these dangerous areas would be included in the obstacles detected by the OGT so the OGT only contributes the semantic information.

Figure 4.18 shows some of the potentially dangerous areas detected by the OGT.



Figure 4.18: Results of detection of potentially dangerous area on a home environment. The first row depicts color images as captured by the mobile robot. The second row shows the potentially dangerous areas as returned by the OGT algorithm. Spheres are drawn for visualization purposes.

4.3.5 Conclusions

In this section, the integration of a domestic robot in an ambient assisting living environment is proposed. The AAL system, that features an RGB-D camera, is able to detect dangerous events such as a person falling or perimeter breaches. Nonetheless, there are small and occluded potentially dangerous areas that cannot be detected with its camera. So in order to enhance the AAL capabilities we propose the utilization of a mobile robot. In this case, a Pepper robot is in charge of detecting small,

non-static potentially dangerous areas, such as near wall outlets, a robotic vacuum cleaner or knives. The position of these objects are forwarded to the AAL so it can consider more dangerous areas.

We put to test out system in different environments with high success. The exhaustive experimentation supports the high accuracy and applicability of the system.

Regarding the limitations of our approach, we realized that the robot tends to accumulate error in the localization process where the features of the location it is at are monotonous. This is due to the way we simulate a laser sensor for the SLAM algorithms. As reported in Section 1.2.2.1, the precise version of our Pepper robot has an error-prone depth camera due to a design fault. As a result, the depth data shows high levels of aberrance, thus providing erroneous measures and eventually causing localization errors.

Regarding future research lines, we are exploring the use of Convolutional Pose Machines (CPM) [Wei et al., 2016, Cao et al., 2017, Simon et al., 2017] to identify the 3D position of a person's joints and thus to track their 3D skeleton. CPM techniques have shown good performance in the detection of articulated objects. First, the joints are located in the color images and second, their 3D positions are estimated taking into account the depth information from RGB-D sensor. This 3D skeleton tracking opens the door to a finer detection of dangerous situations, as arms, head and legs are estimated separately.

4.4 Mapping, navigation and planning system for a social robot with semantic localization

In this section, we propose a mapping, navigation and planning system that can be included in a social robot, Pepper robot in this case, that would assist patients in their daily tasks at home. For this purpose, use will be made of the semantic localisation system developed at [Cruz et al., 2018b]. It will generate a graph representation of the environment from the semantic labelling system and represent its connections in the form of rules with an associated cost. When the robot needs to move between different rooms in the environment, it will consult the route planning system that will return the optimal route, which can be updated in case the connections between rooms have changed dynamically (if there is a closed door or obstacle preventing passage).

So, the main contribution of this work is:

- A navigation system which consists of a mix of semantic localization methods and traditional SLAM.

4.4.1 Semantic localization system

The aim of the Semantic Localization System (SLS) is to compute the location at semantic level. To do this, we based the SLS on the work presented in [Cruz et al., 2018b]. In this work, an optimal methodology for a mobile robot to adapt its knowledge to new environments was proposed. This module works in the following way.

First, the robot captures images of the environment and tries to classify them using an initial pre-trained model. The images used to train this model come from unknown and different homes. As it is deployed in a new environment, it is likely the system obtains low accuracy rates. This is due to the different visual features of the new environment and the environments in which the model was trained in first place. In this case, we can provide information to the robot to collect data and re-identify the locations. In case the category provided by the user is not considered so far by the model, it will be added as a new category. This way, the robot

can easily increase and fit its knowledge to the new environment.

It is worth noting that the model fitting of the SLS is performed before the robot is actually deployed, so it can precisely locate itself once deployed without the interference of the patient.

To achieve this goal, we use the architecture showed in Figure 4.19. This works as follows, an input image is forwarded to the ResLoc CNN architecture. This is a classic CNN architecture which last fully convolutional layer was removed so the output are the visual features descriptor for the input image. As a result, the output of ResLoc CNN is a 2,048 dimensions feature vector.

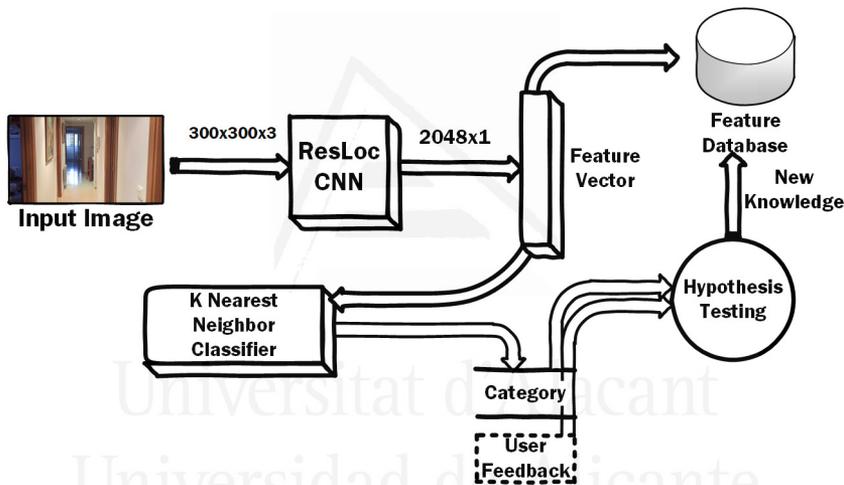


Figure 4.19: This Architecture uses the features of a ResLoc CNN with a vector of 2,048 features as output. The training samples are forwarded to the ResLoc CNN in order to extract their feature vector. The feature vectors construct the model of a KNN classifier.

The visual features and the correspondent categories of each image of the training dataset are extracted using the ResLoc CNN part of the architecture and inserted on the features database. This features database is a model that stores the learned data, which are the features of the training samples. This model is used during the inference stage.

On inference stage, the unknown image is forwarded to ResLoc CNN in order to extract the visual feature vector. Then, a KNN classifier performs a query on the feature database using the recently computed feature vector.

Next, a polling is carried out among the categories of the neighbors, and the most voted category is returned as the final classification of the unknown image.

The performance of the KNN is highly dependent of the k parameter (number of neighbors). Experimentation on this matter is carried out to set the best performing k . We used the Annoy [Annoy, 2018] implementation of the (approximate) KNN classifier.

Then, the model is specialised even more with samples that come for the actual house in which is deployed. The new samples are inserted only if the location fails in a certain room.

As a consequence of this method, the SLS is always updating its model to prevent loosing performance, thus, adapting it as the time goes past. This is specially useful as the appearance of the environment is inevitably going to change. For instance, the furniture is eventually being changed or rearranged, the walls are being painted of another colour, or the home appliances are being replaced.

4.4.2 Motion planning system

When a new location goal is determined as a consequence of triggering a task that must be performed on another room, we need a system that calculates the path from the current room to the target. This task can be done using a simplified map of the environment and an expert system that computes all the paths between the actual location room and the destination room using some connection rules and facts. We named this system the Motion Planning System (MPS).

The map is modelled as a graph where the nodes are rooms, doors and intersections between location access, and the edges are the connection between these places. These connections have an associated direction of movement for the transition (north, east, south and west) and a travel cost that represents the distance between the nodes. Also, we define the transition matrix between node types, where we represent the action that must be performed going from node type A to B , as shown in Table 4.1. Cross type is a node outside a tagged room where various ways join. Interior type is a node inside a tagged room where various ways join. The actions

are: cross the door (cd), follow the corridor (fc) and ND (not defined).

Table 4.1: Transition matrix between node types.

<i>From / To</i>	room	door	cross	interior
room	cd	cd	ND	fc
door	cd	fc	fc	cd
cross	ND	fc	fc	ND
interior	fc	cd	ND	fc

The expert system that computes the paths between nodes has been developed in Prolog, a logic programming language that provides great tools for this kind of task, such as declarative rules and unification (for restrictions management) and backtracking (for graph exploration).

The knowledge is divided into two separated files: facts and rules. The facts are specific to the concrete environment that is being modeled. They contain the definition of the nodes with their types (rooms, doors, cross and interiors), the connection between nodes with the direction and the associated movement cost, and a dynamic predicate that indicates if a door is closed, which can be modified at runtime with the information provided by the robot sensors.

The rules are common to every environment modeled in this way. They check the connectivity between the nodes (direct or indirect) and calculate the path, set of directions, set of actions and the cost from node *A* to *B*. In facts files, the direct connectivity between nodes is represented only in one direction, so we have defined rules that allow the reverse computation of this connectivity, looking for connections from node *A* to *B* and from node *B* to *A* reversing the direction of the movement, as shown in Source Code 4.1. The same principle has been applied to the computation of the actions, as shown in Source Code 4.2. The predicate `action/3` is the Prolog representation of the transition matrix shown in Table 4.1.

Source Code 4.1: Definition of connection rules.

```

0  %Directions
1  dir(north).
2  dir(south).
3  dir(east).
4  dir(west).

5  %Reversibility of orientations
6  revDir(east,west).
7  revDir(north,south).

8  %Revert orientations
9  isRevDir(X,Y) :- revDir(X,Y).
10 isRevDir(X,Y) :- revDir(Y,X).

11 % Look for direct connections
12 hasConnection(X,X, none, 0).
13 hasConnection(X,Y,Direction,Cost) :- dir(Direction),
14     connection(X,Y,Direction,Cost).
15 hasConnection(X,Y,Direction,Cost) :- dir(Direction),
16     isRevDir(Direction,Reversed),
17     connection(Y,X,Reversed,Cost).

```

Source Code 4.2: Definition of action rules.

```

0  %Action definition
1  action(X,X, none) :- isPlace(X).
2  action(X,Y, cd) :- room(X), room(Y), X\=Y.
3  action(X,Y, fc) :- cross(X), cross(Y), X\=Y.
4  action(X,Y, fc) :- door(X), door(Y), X\=Y.
5  action(X,Y, fc) :- interior(X), interior(Y), X\=Y.
6  action(X,Y, cd) :- room(X), door(Y), not(closed(Y)), X\=Y.
7  action(X,Y, fc) :- room(X), interior(Y), X\=Y.
8  action(X,Y, fc) :- cross(X), door(Y), X\=Y.
9  action(X,Y, cd) :- door(X), not(closed(X)), interior(Y), X\=Y.

```

```
10 %Know if X is an existing place
11 isPlace(X) :- room(X).
12 isPlace(X) :- door(X).
13 isPlace(X) :- interior(X).
14 isPlace(X) :- cross(X).

15 %Action rules than ensures reversibility
16 isAction(X,X,Action) :- action(X,X,Action).
17 isAction(X,Y,Action) :- action(X,Y,Action), X\=Y, !.
18 isAction(X,Y,Action) :- action(Y,X,Action), X\=Y, !.
```

For the building of the paths between the nodes, we recursively search for those that are directly connected to the current one until we reach the final node. We have to notice that we are looking for paths without loops (trees), so we do not allow the repetition of any node in them. Without this restriction, the computation of this exploration would hang and enter an infinite loop.

Due to the flexibility of the Prolog module, we can not only calculate the paths between the defined nodes A and B but we can make much more queries, like discover all the accessible nodes from every node, using the same facts and rules knowledge.

4.4.3 Navigation and mapping

We also relied the ROS framework for the complete control of the movement of the robot. This framework provides utilities to interact with the robot and the mapping and navigation methods we adopted for our system.

The task of moving the robot from its current location to another part of the house requires a list of waypoints, which corresponds to the labels (one per room) that are used by the SLS and the MPS and the correspondent locations in the robot coordinate frame. When a task is triggered in a different location in which is intended to be carried out on, the robot guides the patient to the intended location. To do so, the robot localizes itself using the SLS. As a result, a semantic label is obtained.

Then, the semantic label is looked up in the waypoints list. This way the robot is approximately located. Next, the MPS is used to build a plan from the current location to the final destination. This plan is a list of waypoints the robot will try to reach one by one.

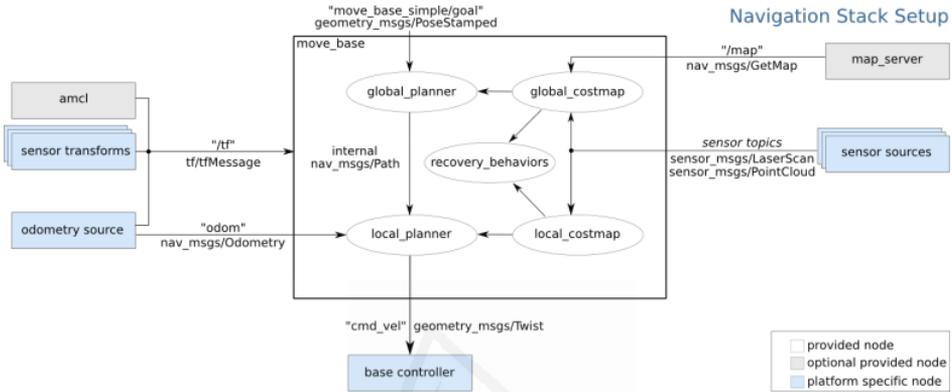


Figure 4.20: Global scheme of the Navigation stack. Extracted from [ROS, 2018]

We relied in the *gmapping* ROS package to create the MPS waypoint list. This method reads the data provided by the laser and creates an occupancy grid map using it. The algorithms that comprehend *gmapping* are thoroughly explained in [Grisetti et al., 2007]. Despite being intended to be used by a laser sensor, we have not used the integrated laser sensors of Pepper because of the lack of resolution, so we adopted the depth image to fake laser readings, as stated in [Perera et al., 2017].

The output of this step is a static 2D map that defines the limits where the robot can move, the walls, the doors and the architectural barriers, but not the moving obstacles. In this map, we define the pose of the waypoints of the MPS, so we can translate the semantic locations to physical positions.

Once we have build the map of the environment, we can load and use it to perform the navigation. Then, we need to determine the position and the orientation of the robot within the map every time it moves. To do this, we used the Monte Carlo Localization, implemented in the Adaptive Monte Carlo Localization (AMCL) ROS package, which is explained in [Fox et al., 1999]. This method samples a set of particles in each itera-

tion that represents a set of probable current poses of the robot. It uses the information provided by the sensors to determine the validity of that prediction and concentrates the next predictions around the older most probably ones.

Finally, we have to resolve the path planning between the current pose of the robot and the target position and move the robot to the goal. This task is done using the ROS *move_base* package, one of the main elements in the ROS Navigation Stack. It receives a goal pose as input, then communicates with components such as the global and local planners, recovery behaviors and costmaps, and generates a velocity command for the base of the robot until it reaches the desired position. The components used by this node are explained in [ROS, 2018] and in the related links of that page.

It is worth noting that we used this mix of semantic localization and traditional mapping and navigation systems because the pure SLAM techniques tend to loose performance on the long term. It is very likely that the robot will not recover the localization once lost despite the accurate state of the art SLAM methods. However, SLS provides an even more accurate localization method because it is based on visual features instead of laser features or odometry, which are often insufficient to provide a robust localization over time.

4.4.4 Experimentation, results and discussion

It is required to build the initial model of the SLS and the corresponding map for the MPS. Figure 4.21 shows the plan of the test house with their rooms. Note that the navigation system will not use the full map but the waypoints in order to set the next navigation goal. The SLS subsystem will provide a good approximate localization for the navigation step.

In the following subsections, we provide experimentation for the MPS, navigation and mapping.

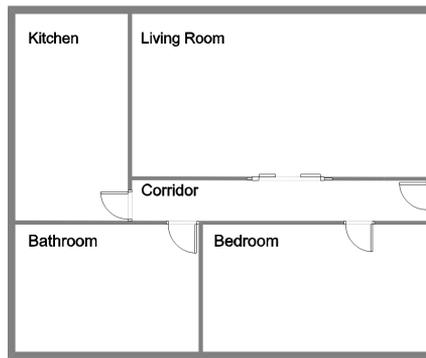


Figure 4.21: The actual plan of a test house.

4.4.4.1 Motion planning system experimentation

For the experimentation of the Motion Planning System we are using the concrete example of the house described in Figure 4.21, with the numerated nodes shown in Figure 4.22. We have defined the node types of every point in Source Code 4.3 so that every line number between 1 and 12 corresponds to the definition of the same numerated node. We have defined the connection between nodes in Source Code 4.4 with the associated cost and the direction that the robot must take to go from node A to B . As stated in Section 4.4.2, the definition of the connection between nodes is only made one-sided.

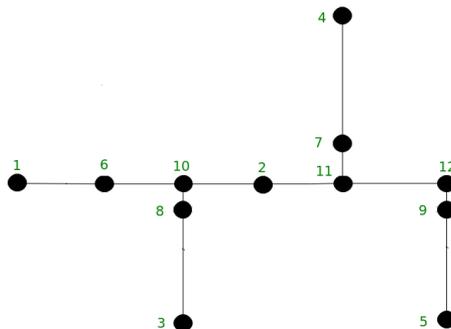


Figure 4.22: Numbered nodes of the house's graph representation.

Source Code 4.3: Definition of node types facts.

```

0  %Type definitions
1  room(kitchen).
2  room(corridor).
3  room(bathroom).
4  room(living_room).
5  room-bedroom).
6  door(door_one).
7  door(door_two).
8  door(door_three).
9  door(door_four).
10 interior(interior_corridor_one).
11 interior(interior_corridor_two).
12 interior(interior_corridor_three).
13 %Types without examples
14 :-dynamic
15     cross/1.
16 %Dynamic predicate to indicate closed doors
17 :-dynamic
18     closed/1.

```

Source Code 4.4: Definition of node connections facts.

```

0  % Connection between nodes
1  connection(kitchen,door_one,east,100).
2  connection(door_one,interior_corridor_one,east,100).
3  connection(interior_corridor_one,door_three,south,100).
4  connection(door_three,bathroom,south,100).
5  connection(interior_corridor_one,corridor,east,100).
6  connection(corridor,interior_corridor_two,east,100).
7  connection(interior_corridor_two,door_two,north,100).
8  connection(door_two,living_room,north,100).
9  connection(interior_corridor_two,interior_corridor_three,east,100).
10 connection(interior_corridor_three,door_four,south,100).
11 connection(door_four,bedroom,south,100).

```

First of all, we calculate the paths from the kitchen to the others rooms,

covering all the existing nodes in the graph. The results indicate that every node can be reached. To ensure the reliability of the rules that grant reverse connections, we calculate the reverse paths of the previous queries too.

Once we have checked that this system calculates all the paths and their reverses, we test the functionality of the dynamic predicate `closed/1`, so we can not reach a goal following a path if there is some door closed.

We have covered all the possible paths between nodes in our experimentation, as shown in Execution Results 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7.

Execution Result 4.1: Execution result from kitchen to bedroom.

```
?- goToFrom(kitchen,bedroom, Path, Actions, Directions,  
  ↪ Cost).  
Path = [kitchen, door_one, interior_corridor_one, corridor,  
  ↪ interior_corridor_two, interior_corridor_three,  
  ↪ door_four, bedroom],  
Actions = [cd, cd, fc, fc, fc, cd, cd],  
Directions = [east, east, east, east, east, south, south],  
Cost = 700.
```

Execution Result 4.2: Execution result from kitchen to bathroom.

```
?- goToFrom(kitchen,bathroom, Path, Actions, Directions,  
  ↪ Cost).  
Path = [kitchen, door_one, interior_corridor_one, door_three,  
  ↪ bathroom],  
Actions = [cd, cd, cd, cd],  
Directions = [east, east, south, south],  
D = 400 .
```

4.4.4.2 Navigation and mapping experimentation

As stated in Section 4.4.3, we have used the *gmapping* algorithm in order to build the 2D static map of the environment, using a fake laser read from the depth sensor of Pepper. The results are shown in Figure

Execution Result 4.3: Execution result from kitchen to living room.

```
?- goToFrom(kitchen,living_room, Path, Actions, Directions,
  → Cost).
Path = [kitchen, door_one, interior_corridor_one, corridor,
  → interior_corridor_two, door_two, living_room],
Actions = [cd, cd, fc, fc, cd, cd],
Directions = [east, east, east, east, north, north],
Cost = 600 .
```

Execution Result 4.4: Execution result from bedroom to kitchen.

```
?- goToFrom(bedroom, kitchen, Path, Actions, Directions,
  → Cost).
Path = [bedroom, door_four, interior_corridor_three,
  → interior_corridor_two, corridor, interior_corridor_one,
  → door_one, kitchen],
Actions = [cd, cd, fc, fc, fc, cd, cd],
Directions = [north, north, west, west, west, west, west],
Cost = 700 .
```

Execution Result 4.5: Execution result from bathroom to kitchen.

```
?- goToFrom(bathroom, kitchen, Path, Actions, Directions,
  → Cost).
Path = [bathroom, door_three, interior_corridor_one,
  → door_one, kitchen],
Actions = [cd, cd, cd, cd],
Directions = [north, north, west, west],
Cost = 400 .
```

4.23. Additionally to the generated map, we have defined the position of the MPS nodes in order to associate a physical location to the semantic ones, so the robot can perform the navigation between the nodes.

Using this map, Pepper can perform the localization using the aforementioned *Adaptive Monte Carlo Localization* with its laser reads. As

Execution Result 4.6: Execution result from living room to kitchen.

```
?- goToFrom(living_room, kitchen, Path, Actions, Directions,  
  ↪ Cost).  
Path = [living_room, door_two, interior_corridor_two,  
  ↪ corridor, interior_corridor_one, door_one, kitchen],  
Actions = [cd, cd, fc, fc, cd, cd],  
Directions = [south, south, west, west, west, west],  
Cost = 600 .
```

Execution Result 4.7: Execution result closing doors.

```
?- closeDoor(door_one).  
true.  
?- goToFrom(living_room, kitchen, Path, Actions, Directions,  
  ↪ Cost).  
false.  
?- openDoor(door_one).  
true.  
?- goToFrom(living_room, kitchen, Path, Actions, Directions,  
  ↪ Cost).  
Path = [living_room, door_two, interior_corridor_two,  
  ↪ corridor, interior_corridor_one, door_one, kitchen],  
Actions = [cd, cd, fc, fc, cd, cd],  
Directions = [south, south, west, west, west, west],  
Cost = 600 .
```

depicted in Figure 4.24, we can see the particles sampled by this algorithm with the most probably poses of the robot. When a well identifiable location is captured by the laser, the density of the particles concentrates over its actual position.

The navigation has been successfully performed with the ROS *move_base* package. The costmaps generated by the planners according to the laser reads locate the dynamic obstacles and let them to compute the optimal path between the current pose and the goal. Additionally, Pepper incorporates an extra level of collision avoiding that blocks the movement of its

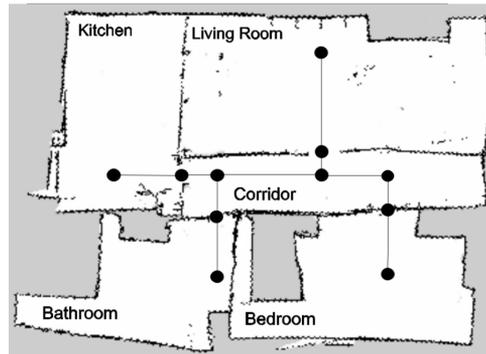


Figure 4.23: The image shows the actual map generated for the MPS with the correspondent graph superimposed. The room names corresponds to the semantic labels used by the SLS. The map was iteratively generated by *gmapping*.

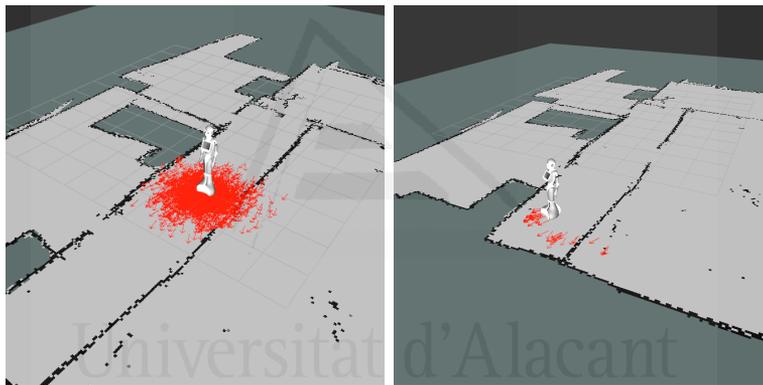


Figure 4.24: Adaptive Monte Carlo Localization running. Every arrow represents a particle with a estimated 2D pose of the robot. **The leftmost image shows high uncertainty so there are multiple plausible poses of the robot (depicted as a big cloud of red arrows around the robot). In the rightmost image, the robot saw a feature that helped to reduce the uncertainty so the plausible poses are significantly reduced (shown as small clusters of red arrows around the robot).**

base when the sonar detects an obstacle. This security margin from the Pepper integrated system makes door crossing difficult when the door is not quite big.

4.4.4.3 Conclusions and limitations

This section has presented a location, navigation and route planning system that makes use of information from a semantic location system.

Semantic labelling provides us with a semantic map of the environment with the different rooms found in the building, information that is suitably complemented with a metric mapping that includes information on the walls and obstacles in each room.

Experiments have shown that the planner is able to properly generate the best route between two locations in the environment from a set of rules that define the connections between the rooms. The system also calculates the actions and the direction of movement to be followed to accomplish the task.

Nonetheless, the system has some limitations. First, the initial stage where the map is created and the models are trained is mandatory and must be carried out by experts, s unsupervised semantic mapping can lead to too many clusters that do not correspond to rooms. On the other hand, although the planning system has dynamic rules to adapt to the changing environment, the mapping does not yet provide for adaptation to changing environments with moving obstacles.

4.5 Fusion of Pepper robot and estimated depth maps method for improved 3D perception

In Section 1.2.2.1, we presented the problem with the Pepper's depth sensor, which generated noisy, distorted and artifact-filled point clouds.

In addition to this specific issue, a variety of other problems can also affect all time-of-flight and structured light cameras. For instance, these sensors provide low density point clouds, or fail on specular surfaces.

In this section, we propose a method to improve the point clouds provided by the Pepper robot v1.8a but which can be used to enhance all point cloud-based cameras. Our proposal is able to provide higher density point clouds and fill the depth information of specular surfaces by involving a deep learning approach for depth estimation. A set of benchmarks validate the improvement of our approach over the depth maps provided by a Pepper robot v1.8a.

The main contributions of this chapter are the following:

- Full description of Pepper's 3D camera issues.
- Use of depth prediction from deep learning networks to complement and correct depth images captured by 3D sensors.
- Algorithm to fuse Pepper's with depth-estimated pointclouds.

4.5.1 The issue with the 3D sensor of the Pepper robot

In order to quantitatively evaluate the distortion of the depth camera of the Pepper Robot, a simple experiment was performed. We took a flat object and placed it at different distances from the 3D sensor. The distances varied between 1 and 3 meters, with an increase in distance of 0.5 meters. For each case, the corresponding depth map was captured and projected into 3D space. All the points on the flat object were selected manually. Depth maps and colored frames were recorded, making this step simple. We next fitted a plane using RANSAC [Fischler and Bolles, 1981], setting the inlier threshold to infinite. This process was done to ensure that all points were unstable values. We tested 60000 different planes were

tested, but only the one with the lowest Root Mean Square Error (RMSE) was returned, this was done because of the random nature of RANSAC. The results of these experiments can be seen in Table 4.2, where the mean Euclidean distance from each point to the estimated plane is reported for each experiment.

Distance (m)	1.0	1.5	2.0	2.5	3.0
Error (cm)	20.36	42.93	65.74	71.80	85.01

Table 4.2: Mean error per distance to the plane reached by the robot.

With the results obtained, we can affirm that the mean error grows as the distance to the object increases. In other words, the quality of the resulting point clouds gradually worsens. In absolute terms, the 3D representation of the planar object is reduced even when the object is as close as 1 meter. In this case, the mean error is 20.36 mm, but the furthest point is 79.15mm from the plane. The furthest point in the 3-meter experiment is 279.12mm. Methodological details related to Pepper and the version of NaoQi used in this experiment can be found in Section 4.5.7. Figure 4.25 shows the point clouds and the corresponding planes estimated for 1 and 3 meters for qualitative assessment purposes.

4.5.2 Other issues related to 3D cameras

There are other problems that commonly affect all sensors based on time-of-flight and structured light. These technologies are used by most commercial depth cameras, such as Microsoft Kinect and Asus Xtion. The first issue we discuss is the impossibility of calculating the depth in shadow that foreground objects project onto background objects. This case produces some areas without depth enforcing around the boundaries of the objects in the scene. The Figure 4.26 Shown, this case. In there is no in the shadow of a conical form that projects the chair in the foreground.

Incompatibility with specular surfaces is another issue. Objects with specular surfaces are not properly sensed by these devices. They produce faulty depth values or areas with no depth information. This case is shown in Figure 4.26. As can be see, the depicted point cloud shows a hole in the

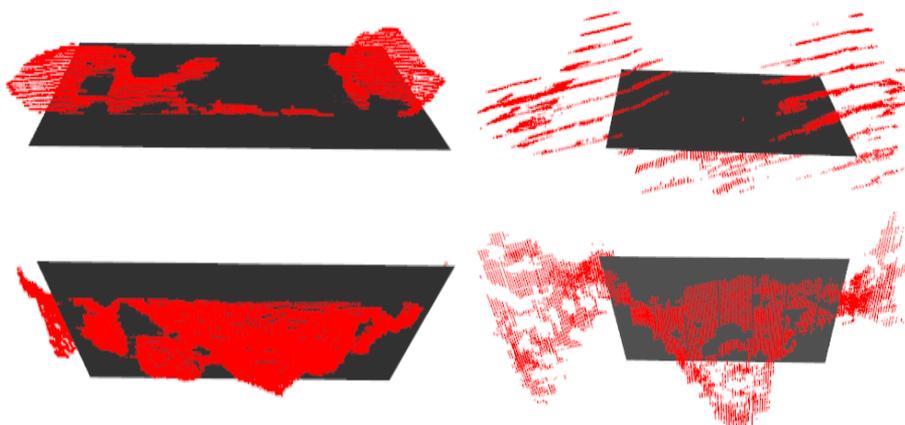


Figure 4.25: Estimated planes (in gray) for the point cloud representations (in red) of a planar object. The first column is the same planar object located at 1 meter from the sensor and the second column is the same planar object located at 3.5 Meters. It is worth mentioning that the representation is far from accurate in both cases, and the problem becomes worse as the distance increases.

place of the specular object.



Figure 4.26: The leftmost image depicts the shadow effect issue of the structured light and time-of-flight sensors. The rightmost image shows another problem with these sensors, which is the incompatibility with specular surfaces.

4.5.3 Issues related to monocular depth estimation

For the proposed approach, we rely on the use of the network proposed by Iro Laina in 2016 [Laina et al., 2016]. As can be found in some of the more recent works [Godard et al., 2017][Atapour-Abarghouei and Breckon, 2018], the comparison with the different novel methods shows that the network by Laina is still among the ones with the best performance, which is why it is considered that this network is good for this approach.

The monocular depth estimation obtained by the Iro Laina network has two main advantages. First, it provides estimations for every pixel on the image, so it generates a very dense point cloud, and usually keeps the geometry of the entities present in the scene.

Nevertheless, it produces several artifacts on the resulting estimation. As shown in Figure 4.27, a trailing artifact is generated near the edges of the objects in the foreground pointing to the background.

We tested the accuracy of the scale and depth values by carrying out the following experiment. We took images of a planar object with the sensor located at different distances. We then estimated the depth maps using Iro Laina's approach and computed the RMSE between the actual distance and the obtained by the sensor. The results of this experiment are shown in Figure 4.28.

4.5.4 FusionV1: Fusing the output of a depth estimation from monocular frames system and the Pepper depth maps

Once we obtained the depth map from the Pepper Robot and the estimated depth map by the aforementioned method, both data are summarized. The summarization is performed as follows:

1. **Look for NaN points in Pepper point cloud.** Find invalid points in Pepper point cloud and save their indexes for further processing. Those points correspond to loss of depth information produced by the aforementioned shadow effect and specular surfaces.

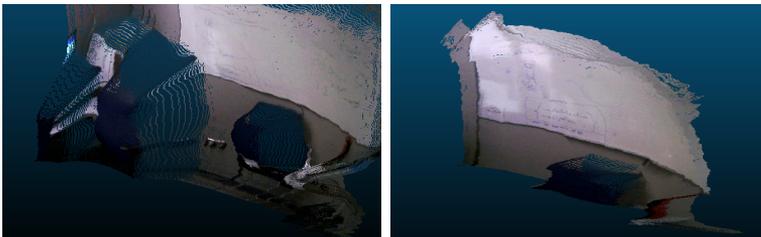


Figure 4.27: Iro Laina's approach produces trailing artifacts from the edges of the objects pointing to the background

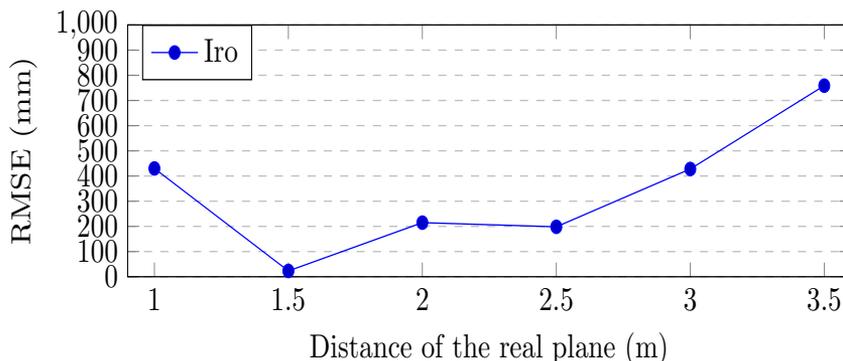


Figure 4.28: RMSE of the distance to the real plane of the points inferred by Iro Laina.

2. **Compare points distances between Pepper and Predicted point clouds.** Each point of the Pepper point cloud is compared with the Predicted point cloud and the distances are computed. This process is straight forward as both points clouds are registered. If the distance is over a threshold Te , the point of the Pepper point cloud is inserted in the Fused point cloud. We do that in order to avoid worsen the Fused output with erroneous predictions.
3. **Calculate the planarity of the remaining points in monocular estimation.** For each remaining point, we calculate a planarity measure. First, we search its neighbor points within a radial threshold r . Then, we approximate a plane using RANSAC, which is robust against outliers. Finally, we calculate the ratio of neighbor points whose distance to the plane is lower than a certain threshold d . This is the planarity measure. If this measure is higher than a determined value Tp , then this information is inserted in the output Fused point cloud. Otherwise, the output information is taken from Pepper point cloud.
4. **Fill the gaps in the output point cloud.** Using the indexes we took apart in the first step, we pick the corresponding points in the Predicted point cloud and check if they are consistent with their neighbors within a certain range in the output ro . If the median of

the distance with their neighbors is lower than a certain threshold m , then we add this information to the resultant Fused point cloud.

The result is a corrected point cloud where the planes are enhanced and the density preserved due to the utilization of the Predicted point cloud, while we avoid the artifacts and isolated erroneous depths produced by the Predicted point cloud as much as possible.

4.5.5 Issues related to the previous fusion method

In the previous section, we proposed a method to fuse the depth information from Pepper's and Iro Laina's approaches. Despite the fact it solved the trailing artifacts contributed by the monocular depth estimation, our method, hereinafter referred to as *FusionV1*, establishes a very restrictive filter and rejected a large quantity of points, as depicted in Figure 4.29. As a result, the method provided low-density point clouds. In addition, FusionV1 preserves the poorly accuracy of the depth predictions of Iro Laina's approach as shown in Figure 4.30.

4.5.6 FusionV2: Refining the fusion of monocular and Pepper depth maps

In order to solve the main issues of FusionV1, we propose changes to this previous method, taking the noisy and incorrect point clouds from Pepper and the point clouds as provided by Iro Laina's method, and returning a new corrected point cloud.

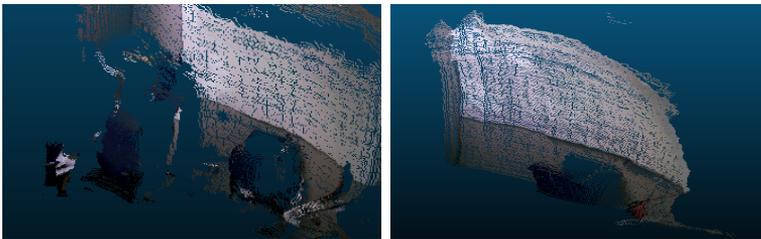


Figure 4.29: FusionV1 provided poor density point clouds as a result of a strict filtering process.

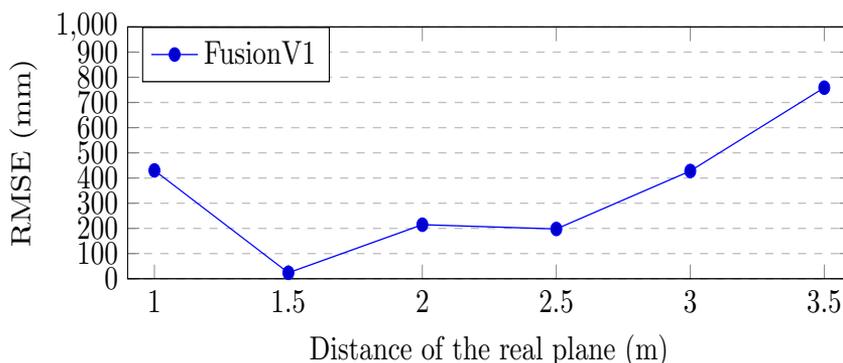


Figure 4.30: RMSE of the distance to the real plane of the points fused by FusionV1.

First, we obtain the raw point cloud from Pepper and conduct no pre-processing.

Then, Iro Laina’s approach [Laina et al., 2016] estimates the depth map from a monocular frame using a fully convolutional neural network. Then, both point clouds are summarized. The fusion process consists of mixing the point cloud from Pepper’s camera and that one estimated from the monocular color image. This process is carried out as follows:

1. **Aligning the monocular depth estimation with Pepper’s point cloud.** First, we want to align both point clouds. This step would be computationally expensive if we use the whole points, so we perform uniform sampling on the Pepper’s point cloud and obtain the corresponding points from the monocular cloud (this step is straightforward because both are registered and obtained from a depth map of the same resolution). Then, we align the point cloud estimated with Iro Laina’s approach with Pepper’s using a variant of the Iterative Closest Point algorithm that employs the Single Value Decomposition technique. This method not only calculates translation and rotation between both point clouds but the scale component is also inferred [Du et al., 2007]. As a result, the estimated point cloud with Iro Laina’s method is set in the correct scale.
2. **Calculating the planarity of every point in monocular es-**

timation. We calculate a planarity measure for each point in Iro Laina’s method estimated point cloud. First, we search for its neighboring points in the same point cloud using a radial search within a range threshold r . We then use RANSAC to fit a plane using these points. This method is robust against outliers and calculates the plane that better represents the data. Next, the ratio of neighboring points whose distance to the plane is less than a certain threshold of d is calculated. This ratio is called *planarity measure*. If this ratio is higher than a threshold of Tp , then the monocular estimation point is inserted in the cloud-fused output points. If not higher than the threshold, the output information is taken from the Pepper point cloud.

The resultant point cloud has the density and geometric shapes of the point clouds estimated by Iro Laina’s method and preserves the correct depth values and scale provided by Pepper. This approach is depicted in Figure 4.31.

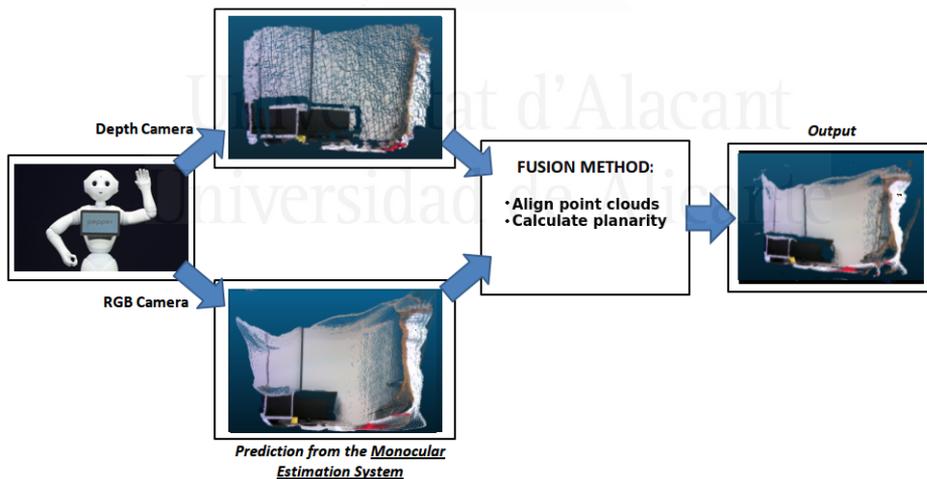


Figure 4.31: Scheme of the new fusion method FusionV2.

4.5.7 Experimentation

The experiments were carried out using version 1.8a of the Pepper Robot, which includes an Asus Xtion as a 3D sensor. The manufacturer

provides a set of parameters for the camera, so it is not necessary to estimate the intrinsic parameters. The center of the image in the X-axis is 319.5, and the center of the image in the Y-axis is 239.5. Its focal length is 525. With these parameters, we can project the provided depth maps into 3D point clouds.

For the computation of the fusion method and the monocular depth estimation, we used an external computer with 8 GiB HyperX DDR3 RAM (Kingston) 1600 MHz on an Asus P8H77-M PRO, also including an Intel Core i5-3570 processor and an NVIDIA Quadro P6000 GPU. To execute the DL tasks, we used TensorFlow 1.8 as the core with Keras 1.2.0. The OS used was Ubuntu 16.04. To accelerate the computations, we used CUDA 9.0 and cuDNN v7.5.

To ensure the reliability of our proposal, we performed a set of experiments. With Pepper's color camera and depth sensor, we took several captures of a whiteboard of our workplace at different distances, as shown in Figure 4.32.

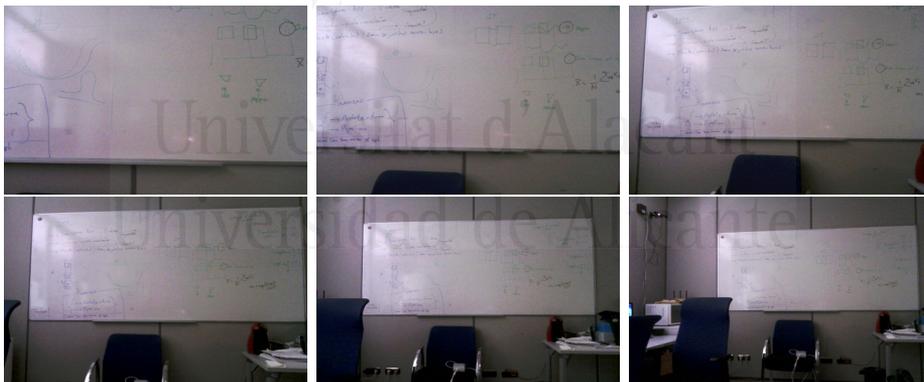


Figure 4.32: Monocular captures of the scenes we used in the experiments.

We tested the point clouds generated by Pepper, Iro Laina's method, our former fusion method *FusionV1*, and our new fusion method *FusionV2*. As the generated point clouds depicted more objects in addition to the planar object, we manually selected the whiteboard planes and removed the remaining points. These resultant point clouds, which are depicted in Figure 4.33, were used for the experiments.

The first metric we tested was *Planarity*. This metric consists of fitting

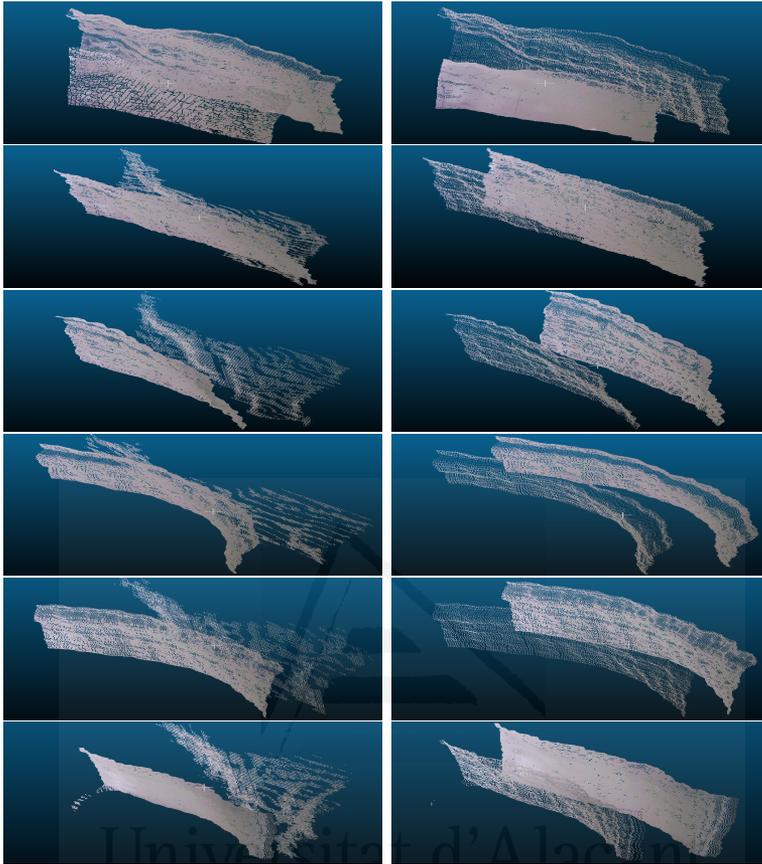


Figure 4.33: Comparison of the generated planes from Pepper, Iro, FusionV1, FusionV2 point clouds. In the left-hand images, Pepper and Iro clouds are compared, while FusionV1 and FusionV2 are compared in the images of the right. This split was made done because Pepper and FusionV2 overlap in distance, and Iro and FusionV1 are quite similar in distance and geometry. In the left-hand images, the least dense cloud corresponds to Pepper. In the right-hand images, the least dense cloud corresponds to FusionV1

the best possible plane for every cloud using RANSAC, and calculating the RMSE of all the points. The results are shown in Table 4.3 and Figure 4.34. As can be seen, the planarity RMSE of the Pepper point clouds worsens as the distance is increased. On the other hand, the planarity RMSE of Iro Laina’s approach and both fusion methods are more dependent on the visual features than on the distance of the objects from the sensors. Nonetheless these methods provide better planar objects.

The second metric is *Distance precision*. This metric consists of deter-

Distance	Pepper	Iro Laina	FusionV1	FusionV2
1	12.422	22.0738	22.2634	15.1578
1.5	23.2205	11.9214	12.3538	12.079
2	37.6301	19.7721	21.2261	22.2253
2.5	56.4277	69.0246	67.4973	73.1071
3	70.0697	62.7262	61.3264	71.8479
3.5	112.202	47.8647	38.4259	53.9838

Table 4.3: Planarity results obtained for the point clouds provided by the different methods at different distances. Distance is in meters whilst the results are in millimeters.

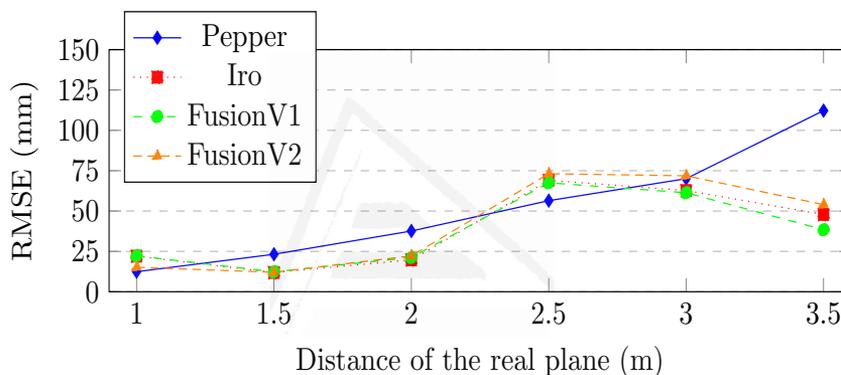


Figure 4.34: RMSE distances of the point clouds to the fitted plane

mining the difference between the estimated distance and the actual one (we have measured the exact distance to the planes with a laser), and calculating the RMSE of all the points. The results are shown in Table 4.4 and Figure 4.35. As depicted in previous figures, the distance RMSE of the Pepper planes is far better than Iro's, which has far more problems with the scale of the cloud. However, *FusionV2* method preserves the good distance estimation of Pepper.

The last metric is *Density*. This metric measures the proportion of good points (not NaNs) with respect to the total number of points in the cloud. The results are shown in Table 4.5. These results show the lack of resolution in Pepper's point cloud and the good quality of Iro's. Our previous method *FusionV1* inherits this bad density in some situations but our new method, *FusionV2*, preserves the density quality of Iro's.

Distance	Pepper	Iro Laina	FusionV1	FusionV2
1	93.7632	430.007	430.114	91.2979
1.5	81.9094	23.0732	23.5829	79.1551
2	55.7695	214.786	214.566	129.712
2.5	70.1055	198.306	197.639	120.525
3	119.266	427.858	427.958	106.923
3.5	285.817	758.839	758.784	345.566

Table 4.4: RMSE distances of the point clouds to the actual plane. Distance is in meters whilst the results are in millimeters.

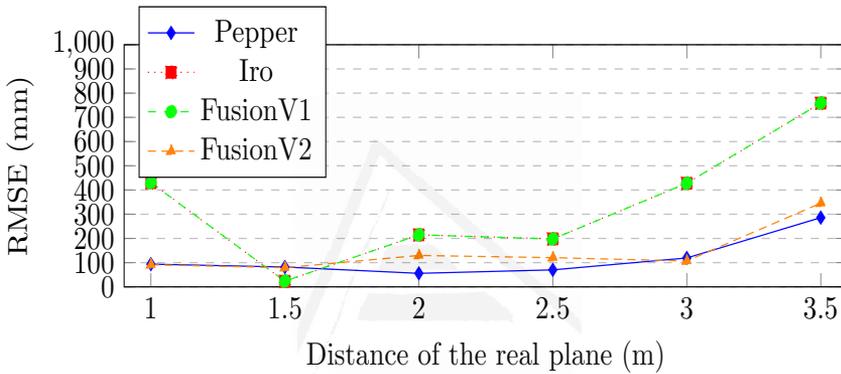


Figure 4.35: RMSE distances to the real plane for this experiment.

Distance (m)	Pepper (%)	Iro (%)	FusionV1 (%)	FusionV2 (%)
1	23.9551	100	23.9551	100
1.5	23.5073	100	39.8207	100
2	23.4122	100	23.4122	100
2.5	23.2482	100	23.2482	100
3	23.2825	100	23.2825	100
3.5	22.9414	100	22.9414	100

Table 4.5: Density of the point clouds.

4.5.8 Conclusions

We have presented a refinement of our previously proposed method for improving the quality of the Pepper 1.8a depth map.

The Pepper camera undergoes a type of radial distortion, arguably produced by its lenses, that which produces several artifacts on planes. In order to correct this geometrical issue, we make use of the monocular

depth estimation of *Iro Laina*'s architecture, which performs better with planar representation.

We fuse the Pepper and the monocular point cloud, aligning the latter with the former and searching for planar areas. Thus, we provide a point cloud of better quality. As shown in the experimentation section, our new fused output point cloud far better represents the distance of the scene than our previous one, and preserves the density of points of the monocular depth estimation.



Universitat d'Alacant
Universidad de Alicante



Universitat d'Alacant
Universidad de Alicante

Investigations applied to social robotics

In this chapter, contributions to social robotics whose subject matter does not fit into any of the previous sections are presented. A brief introduction is provided in Section 5.1. Section 5.2 reviews the state of the art of social robotics with a focus on elderly care and people with autism. Section 5.3 proposes an augmented reality system to perform and evaluate rehabilitation exercises at home. Finally, Section 5.4 presents the results of an investigation that mixes DL techniques and statistical classifiers to achieve a robust system of online learning and recognition of people.

5.1 Introduction

According to the World Health Organization (WHO), about 15% of the world's population suffers some form of disability. In addition, this rate is continuously increasing as a result of the society aging and the growth in the prevalence of chronic diseases such as cancer or mental health disorders. This fact has led to a social concern about their health care, especially those people with significant difficulties in functioning.

In this regard, one of the most noteworthy shortcomings is the rehabilitation services since they play a main role in the person's autonomy reinforcement, the decrease in their vulnerability and the improvement in their

physical condition. Furthermore, a proper recovery prevents an early retirement from work, a considerable decrease in accumulated wealth and/or a reduction of social functions. However, the deficiencies in rehabilitation services, their cost and their long duration demand the development of technology supporting this process at home.

In the following sections, we will make a detailed analysis of the different social robots that have been used throughout the years, focused on the attention to elderly people and people with autism, explaining their characteristics and the way in which they help in therapies. On the other hand, we also propose a physical rehabilitation assistance system based on augmented reality, which allows to evaluate the assessment of therapies in an automated way, and presenting the results so that they can be consulted and monitored by professional therapists. Finally, we propose a person recognition system capable of learning new identities in a few seconds, allowing social robots to interact in a personalised way with the people around them.

5.2 Socially assistive robots for older adults and people with autism

A review of the literature reveals the enormous variety of assistive technology currently available. Given the wide range of types and levels of deficiency, assistive technology can be classified depending on its complexity. Three concentric spheres of assistive technology can be defined with the user at their centre. These are (from the inside to the outside): embodied assistive technology, assistive environments, and assistive robots.

Embodied assistive technology includes mobility devices [Pant et al., 2018, Tahsin et al., 2016] (e.g. wheelchairs, prostheses, exoskeletons or artificial limbs); specialized aids (e.g. hearing [Abdallah and Fayyoubi, 2016], vision [Suresh et al., 2018, Phillips and Proulx, 2018, Bhowmick and Hazarika, 2017], cognition [Palmqvist and Danielsson, 2019] or communication [Davydov and Lozynska, 2016]); and specific hardware, software, and peripherals that assist people with disabilities in accessing information technologies (e.g. computers and mobile devices). Although these systems

provide valued help, they usually offer just one functionality and most lack intelligence (intelligence understood as the ability to receive feedback from the environment and adapt their behaviour).

Going a step further, the environment can be adapted to the user's needs, with sensors and actuators, such as cameras or domotic systems, such that more functionalities are covered and more information about the user's health status can be gathered and processed, providing this technology with intelligence. In this line, we can find the smart homes [de Oliveira et al., 2016], virtual assistants [Escalona et al., 2019, Costa et al., 2018b, Costa et al., 2017] and AAL settings [Ruano et al., 2019, Costa et al., 2016, Gomez-Donoso et al., 2019]. Nevertheless, this kind of technology fails to support independent life when the user has chronic or degenerative limitations in motor and/or cognitive abilities.

As a solution, Assistive Robotics (AR) emerged. Its main goal is to fruitfully promote the well-being and independence of persons with disabilities. Robots may assist people in a wide range of tasks at home (especially in terms of activities for daily living), and so ongoing research includes household robots [Coşar et al., 2020, Andrade-Cetto and Torras, 2006, Cruz et al., 2018a] and rehabilitation robots [Luxton and Riek, 2019, Martinez-Martin and Cazorla, 2019], among others. In the case of assistive robots, interdisciplinarity is required to achieve the final goal, integrating research areas such as AR, Human-Robot Interaction (HRI) and ML techniques, among others.

Thus, motivated by the current societal needs of the particular risk groups (i.e. children and older adults), this section reviews and summarizes the promising and challenging research on AR aimed at helping older persons and children with autism to perform their daily tasks.

5.2.1 Socially Assistive Robots

One of the main difficulties in the acceptance of assistive technology is the way in which this technology is perceived. In this sense, the interaction between the robot and the user is a key issue. This social interaction led to the development of Socially Assistive Robotics (SAR). According to Feil-Seifer and Mataric [Feil-Seifer and Mataric, 2005], SAR can be defined as

the intersection of AR and Socially Interactive Robotics (SIR), whose main task is interaction with human individuals.

Ideally, SAR should operate autonomously and not require the manipulation of a human operator. The interaction with the user must be intuitive and not require extensive training. Additionally, the robots have to adapt their behaviours to the new routines and needs of the users, which is currently the most challenging task to be solved [Tapus and Mataric, 2008]. To meet this demand, Artificial Intelligence (AI) and ML algorithms must be developed and deployed in these systems, since the robots cannot be programmed in advance to react to every possible circumstance that might occur during interaction with the users.

As mentioned above, there exists a wide variety of applications depending on the needs to be covered and the demands of the target social group. Given that the SAR focuses on improving the user's life conditions, this study reviews the advances in two of the most vulnerable social groups:

- *Older adults*
- *People with cognitive disorders*

Section 5.2.2 reviews the latest advances in age-related health issues; while Section 5.2.3 analyses the most significant research on children with autism in terms of diagnosis and therapy to train their communicative and social skills.

5.2.2 Older adult care

The ageing population is today's major health concerns. This unprecedented situation urgently requires technological solutions to confront the constantly increasing demands of care services, which are currently overwhelmed. In this regard, the WHO identifies two key concepts in its *Global strategy and plan of action on ageing and health* [World Health Organization (WHO), 2017]:

- healthy ageing, understood as the process of developing and maintaining functional ability for older people's well-being

- functional ability, where technology is used to perform functions that might otherwise be difficult or impossible

Healthy ageing has become popular topic in recent decades. In this regard, SAR develops systems to improve older people's health through physical activity, which has a positive cognitive impact [Mura and Carta, 2013]. Some research attempts have consisted on companion robots that help users with assisted therapy and activity (see [Martinez-Martin and del Pobil, 2017b] for an overview). However, work is needed to promote for their acceptance among older people, as pointed out in [Oh et al., 2019], especially in terms of social interactions.

In addition, SAR for promoting physical exercise has been developed. This is, for instance, the case of the robotic *coach* proposed by Görer et al. [Görer et al., 2016]. It is essentially a technique based on a learning by imitation approach, which is used to learn the exercises from a human demonstrator. Then, the *reference* joint angles are used to evaluate the user's movements and to provide them with the necessary feedback to improve their performance. Note that two different platforms are used to achieve this goal. A NAO robot is used to describe the physical exercises, while an RGB-D camera captures the movements of the person. This can be problematic since the correct position of the RGB-D device is essential to properly evaluate the user's performance. In addition, no sitting exercises are used because the skeleton data is insufficient to obtain the required results. Finally, the robot may confuse the user, given that it emulates the exercise as a demonstration and performs certain movements that are not to be carried out, such as head motions.

Another proposal is PHAROS [Martinez-Martin et al., 2019, Costa et al., 2018a], a socially assistive robot that monitors and evaluates the daily physical exercise done at the user's home (see Figure 5.1). For this, HRI techniques (i.e. a CNN together with a Recurrent Neural Network (RNN)) are used to properly identify and evaluate the exercise performance. In addition, it integrates a recommender that generates the exercise workout every day such that the person is working on what is necessary to stay healthy.

Assisting functional ability requires more complex systems. In this



Figure 5.1: Pharos robot in a pilot study at the residence for the elderly Doña Rosa (Alicante)

sense, systems have been evolving over time, integrating an increasing number of functionalities. This is the case of the HOBBIT [EU project, 2013], a robot to help older people feel safe and continue to live in their own home. With this aim, the robot, illustrated in Figure 5.2, is able to autonomously navigate around the user's apartment, going anywhere they request, being able to pick up objects from the ground, bring a specific object, learn new objects to be found in the future, call in case of emergency, provide games for entertainment, and also remind the user to take the medication.

Analogously, the EU project RAMCIP [EU project, 2020] has developed a robotic assistant for older adults and those suffering from Mild Cognitive Impairments (MCI) and dementia (see Figure 5.3). This robotic assistant also integrates several functionalities that promote physical and cognitive activity, such as detecting a fall (in which case a relative or external caregiver is informed), checking the cooker has been turned off after



Figure 5.2: Hobbit robot in a pilot study at the Doña Rosa senior care home.

preparing a meal or the lights have been turned on when walking at night, picking up improperly left or fallen objects from the ground and moving them to safe storage, reminding users about their medication, bringing the corresponding medicine and monitoring its taking and facilitating social interaction with family and friends.

Other solutions consider the possibility of integrating a robot platform into a smart home environment such that its functionalities may be augmented. An example is the Robot Activity Support System (RAS) created by the Washington State University [Wilson et al., 2019] for adults with memory problems and other impairments to help them to live independently. Thus, the smart home has sensors in the walls to track the user's movement and feeds their data into the robot's neural network. This allows the robot to integrate activity detection technology to provide assistance when required. However, it is still at an early stage of development and, can only provide video instructions on how to do simple tasks, such as assisting a person through the steps of taking a dog for a walk or guiding them to an object. In addition, the need to install additional technology at home makes this option difficult and costly to implement.

Alternatively, other developments aim to assist people in nursing homes and healthcare facilities. In these kinds of systems, the key issue is the social component, with the aim being for the older adult user to perceive the robotic platform as a social companion rather than a machine to perform predefined tasks. This is the main focus of Rudy [INF Robotics, 2019], an assistive robot created by INF Robotics in 2017. This robot offers telemedicine capabilities, such as Remote Patient Monitoring (RPM), as



Figure 5.3: RAMCIP robot in a pilot study at a user's home



Figure 5.4: Rudy in a pilot study

well as medication reminders and dispensing (shown in Figure 5.4). In addition, it integrates a social component that, together with its friendly appearance, engages users. In fact, the social interactions are the most appreciated functionality of this system since loneliness is a major issue among the ageing. Nevertheless, it costs \$5000, which is a significant amount that is not within all budgets.

In a similar line, Trinity College Dublin developed Stevie in 2017, which they improved in 2019 as Stevie II (Figure 5.5). The aim of this socially assistive IA robot was to augment the role of caregivers in long-term care environments, allowing them to concentrate mainly on person-centred tasks. Its functionalities range from medication reminders to keeping residents cognitively stimulated with quizzes and games. For this, enhanced expressive capabilities and a well-defined social component are used.



Figure 5.5: Stevie II in a pilot study

5.2.3 Training communication and social interaction in children with autism

In recent years, the use of SAR has become popular for the treatment and diagnosis of autism [Dickstein-Fischer et al., 2018]. Indeed, the research in this field has presented an increase in user therapy acceptance and improvements in their social skills [Scassellati et al., 2012].

Applied behaviour analysis (ABA) is one of the most extended therapies for the treatment of autism. It consists in improving specific behaviours, which are divided into simple and repetitive tasks that are presented sequentially and strategically while measuring and analysing the patient's performance during the therapy [Kasari and Lawton, 2010].

The automation of some aspects of the therapy using technology with different devices and tools has been widely studied. Videos, virtual and augmented reality, and robotics [Goldsmith and LeBlanc, 2004]. ABA therapies combined with SAR have exhibited substantial advantages and demonstrated their effectiveness in obtaining positive results in patients, such as high enthusiasm, increased attention and social activity [Begum et al., 2016]. Figure 5.6 shows an example of this interaction.



Figure 5.6: Frames of a video with the interaction between a child with autism and a therapist. It is used to compare the engagement of the child with an activity using a SAR (Pleo). Extracted from [Scassellati et al., 2012].

These results may be explained by the fact that children with autism feel more comfortable interacting with robots, because their behaviour and reactions are more predictable [Scassellati, 2007]. Furthermore, the social skills of the patients could be gradually improved by increasing the complexity and unpredictability of the robot's behaviour, making it more

similar to actual human interaction [Dautenhahn and Werry, 2004].

These robotics systems can be used to manage therapy sessions, collect data and analyse the interactions with the patient, and generate information from this data in the form of reports and graphs. For this reason, they are a powerful tool for the therapist to check patient's progress and facilitate diagnosis.

The visual appeal of the robotics platform is a key factor to engage the attention of children with autism. In general, these robots tend to use bright colours, rotating mechanical parts, striking shapes and lights [Cabibihan et al., 2013]. Additionally, some studies have reported that children with autism prefer to interact with robots with less humanoid characteristics [Robins et al., 2006]. However, some anthropomorphic robots have been successfully used in research, especially in imitation and emotion recognition activities. Tables 5.1 and 5.2 present different SAR robots used in experiments. Following [Ricks and Colton, 2010], there are several robot types depending on their location on the humanoid spectrum:

1. **Android.** Look like humans.
2. **Mascot.** Humanoid form but abstract or cartoonish appearance.
3. **Mechanical.** Humanoid form with visibly mechanical parts.
4. **Animal.** Look like pets.
5. **Non-Humanoid.** No resemblance to any living being.

Since the therapist's availability is limited, SAR must be developed with a certain level of autonomy in order to carry out the treatments. This autonomy is directly correlated with its level of intelligence in adapting to the environment and the patient's responses. This is where ML comes in, providing solutions to the problems these systems must address, such as eye-tracking, face or automatic speech recognition.

Eye-tracking is the process of measuring the point of fixation of the gaze or the movement of an eye with respect to the head. It is used to measure the patient's attention to the robot. There exist commercial solutions for this purpose, but they are high cost or depend on special and

Robot	Appearance	Type	Description	Publications
Zeno R-50		Android	Child-sized robot (height=0.64 m and weight=6.5 kg) with a simplified expressive face. Its face has a motor that can be animated using software.	[Salvador et al., 2015, Salvador et al., 2016, Silva et al., 2017]
Nao		Android	Humanoid (height=0.57 m and weight=5 kg). Appearance of a human toddler. 11 DOF for its lower limbs and 14 DOF for its upper body.	[Geminiani et al., 2019, Chevalier et al., 2016, Lytridis et al., 2018, English et al., 2017, Qidwai et al., 2019, Shamsuddin et al., 2012, Tapus et al., 2012, Petric et al., 2017, Mavadati et al., 2016, Yang et al., 2018]
Pepper		Android	Humanoid (height=1.21 m and width=0.48 m). Almost the same articulations than a human, except for its base and fingers. It has 4 microphones, two loudspeakers, two RGB cameras and a depth sensor (Asus Xtion). Tactile sensors in the head and the back of its hands. Speech recognition engine that is able of identifying multiple variations in the human voice.	[Azuar et al., 2019, Burkhardt et al., 2019, Nunez et al., 2015, Yabuki and Sumi, 2018]
KASPAR		Android	Child-sized humanoid robot with minimal expressions. Body movements and gestures using its hands, arms, torso, head and show facial expressions.	[Wood et al., 2017, Wainer et al., 2014b, Wainer et al., 2014a, Wood et al., 2019, Huijnen et al., 2016, Dautenhahn et al., 2009, Robins et al., 2009]
Keepon		Animal	Small creature-like robot (height=12 cm). Simple, like a yellow snowman, and made of soft materials (silicone rubber).	[Kozima et al., 2009, Kozima et al., 2005, Azmin et al., 2016]
PABI		Animal	Penguin-like small robot. 8 DOF in eyes, head, wings and opening beak. Single board computer for autonomous operation and wireless communication for teleoperation. Speaker mounted behind its beak for communication. 2 independent video cameras in its eyes for tracking and monitoring. It carries a tablet as an interface with the onboard computer.	[Woodyard et al., 2015, Brown, 2018, Dickstein-Fischer et al., 2017]
Pleo		Animal	Dinosaur-like robot. Learn and repeat dances. 14 DOF, with movable legs, torso, neck, eyes, tail and mouth. Touch sensors in its whole body. Camera in its nose for object tracking and microphones. Capability to show emotions by making noises.	[Kim et al., 2013, Kim et al., 2012, Larriba et al., 2016, Curtis et al., 2011]

Table 5.1: Robots used in autism therapies.

Robot	Appearance	Type	Description	Publications
Robota		Android	Small robot (height=45 cm and width=14 cm) with the form of a young girl. 1 DOF of movement in its limbs (up and down), head rotation, 1 DOF for every arm, coordinated motion of the two eyes, individual blinking of the eyes and touch sensibility. Capabilities for vision tracking and ML.	[Dautenhahn and Billard, 2002, Billard et al., 2006, Billard, 2003, Robins et al., 2004a, Robins et al., 2004b]
i-Sobot		Android	Biped robot (height=16.5 cm and weight= 350 g). 17 pieces of micro servo motors for walking and 180 different actions. 180 voice and sound commands. Remote controller or spoken commands.	[Watanabe and Yoneda, 2009, Kaur et al., 2013, Srinivasan et al., 2013]
Tito		Mascot	Robot (height=17 cm) Coloured red, yellow and blue with washable clothes made of soft material. Wheels to move but with fake feet and legs to emulate human shape. Movable arms and head, lighting mouth for smiling. Wireless microphone-camera device inside one eye for tracking. Touch sensibility. Autonomous and teleoperated modes.	[Duquette et al., 2008]
GIPY-1		Mechanical	Cylindrical mobile robot (diameter= 20 cm and height=30 cm). Its face is the cladding of the robot: round eyes and nose triangle, with elliptical mouth. Can move forward, backward and turn on its own axis. Wireless controlled by a joystick.	[Pradel et al., 2010, Giannopulu and Pradel, 2010]
HOAP-3		Android	Humanoid robot (height=60cm and weight= 8.8 kg), commercialized by Fujitsu. 28 DOF for head, arms, legs and body movement. Inbuilt camera in its eyes for tracking and recognition. Microphones and speaker for audio recognition and speech. Expression LEDs to show emotions. Autonomous operation and teleoperated through WIFI.	[Ravindra et al., 2009]
Ifbot		Mascot	Humanoid robot (height= 45 cm). 2 moving arms with 1 DOF and two wheels to move. 10 motors for facial expressions: eyes, eyelids and neck. 104 LEDs in its head and mouth to show emotions along with the facial expression.	[Lee et al., 2012]
Cosmobot		Mascot	Movable head, arm and mouth. Wheels to drive the robot in 4 directions. Pressure sensors and a built-in microphone for the interaction with the children. Expandable play station (Mission Control) for interaction, with external ports for joystick, wearable head and arm sensors. Teleoperated and controllable from a desktop computer software.	[Lathan et al., 2007, Tzafestas, 2016]
Ryan Companionbot		Android	Rear-projected humanoid. 3D avatar models with speech and facial expressions. The animated face is projected into a face-shaped translucent mask. The 3D models are compatible with Maya design software.	[Askari et al., 2018, Askari, 2018, Mollahosseini et al., 2018]

Table 5.2: Robots used in autism therapies.

invasive hardware (Tobii EyeX). However, there are many works focused on inferring the gaze of the users from images of their faces. Traditional techniques usually rely on shape-based methods, such as [Ishikawa, 2004, Valenti et al., 2011], observing geometries like pupil centres and iris edges, or in appearance-based methods, such as [Baluja and Pomerleau, 1994, Sewell and Komogortsev, 2010], which directly use the images of the eyes for the prediction, with handmade features along with neural networks. In recent years, the focus has been on deep learning techniques to accomplish this task using standard and inexpensive camera devices. This is the case of [Krafka et al., 2016], which uses a CNN to predict the gaze of the user from a colour image of their face, previously trained with a large-scale dataset of faces and correlated gazes. More recent works such as [Sims et al., 2019] predict emotions and the patient's mood states from eye tracking data using RNNs.

The study of the patient's gaze is a crucial technique that helps with the diagnosis of autism and measures the effectiveness of the interaction between the robot and the user. In [Damm et al., 2013], the researchers carried out a study comparing the gaze attention of patients with autism when they interacted with humans and with robots, as shown in Figure 5.7. Similar to the previous example, in [Yoshikawa et al., 2019] the authors compare the gaze attention of people with autism while maintaining conversations with a human and a realistic android, which could serve as a diagnostic tool. In [Robins et al., 2004a, Robins et al., 2004b] the authors report the effects of repeated exposure to the humanoid robot Robota, which includes an increase in gaze attention and imitation.

Most of the experiments with these robots do not specify the kind of eye-tracking technique they use, or even whether they use external hardware, but recent works in this topic show that deep learning techniques outperform traditional ones without the need for invasive tools, so developments may move in this direction in order to ensure the best experience for users.

Face recognition has been one of the most widely studied research topics in computer vision since the beginnings of computer science, as it provides the recognition of subjects in a non-intrusive manner. The first step



Figure 5.7: Experiment measuring the interaction of patients with autism with robots and humans using an EyeSeeCam and tracking gloves. Extracted from [Damm et al., 2013]

involves the detection and delimitation of the region of the image containing the face. Traditionally, detection has been conducted by searching for handcrafted features, like in [Viola et al., 2001], which uses cascade classifiers with different resolutions, trained with the Adaboost technique, based on Haar-like features. Subsequently, a vector of characteristics is extracted to describe the face, using global techniques like Eigenfaces [Turk and Pentland, 1991] or Fisherfaces [Belhumeur et al., 1997] based on Principal Component Analysis (PCA), or using local descriptors, like Local Binary Pattern Histograms (LBPH) [Ahonen et al., 2004], which codify the local structure of the image by comparing every pixel with its neighbourhood. However, traditional methods suffer when the conditions of the face are not ideal: recognition rates decrease with variations of the pose of the face and changes in the lighting conditions. Recent works have adopted end-to-end architectures based on deep learning that greatly outperform the traditional methods. Studies such as [Sun et al., 2015, Masi et al., 2018, Yucel et al., 2018, Deng et al., 2019] use variations of CNN architectures trained with large-scale face datasets, obtained without pose restrictions, with good results on tests. Along with face recognition, recent studies like [Pramerdorfer and Kampel, 2016, Kahou et al., 2016], classify the user's emotions by means of variants of CNNs, with promising results.

These characteristics are important for SAR in order to identify the

patient and their mood and keep track of the history of the interaction. In [Azuar et al., 2019], the researchers use face and emotion recognition to make a Pepper robot adapt a story to the mood of the children. In [Ismail et al., 2011], the authors propose a technique for face recognition using a humanoid robot NAO to track the faces of the children with autism and measure their concentration during social interaction. In [Nunez et al., 2015], the authors propose several activities through the interaction with a Pepper robot, receiving feedback by measuring the users' smile, as shown in Figure 5.8.

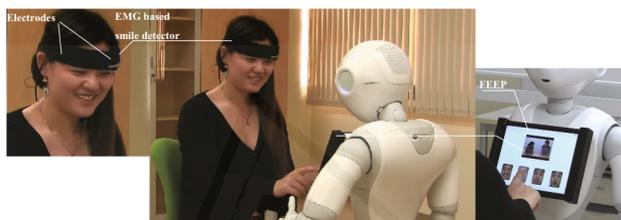


Figure 5.8: Experiments of interaction with patients with autism using smile measurement as feedback. Extracted from [Nunez et al., 2015]

Finally, *automatic speech recognition* is considered the most important bridge to enable human-machine communication. However, the technical difficulties of speech processing led to the keyboard and mouse becoming the most accurate interfaces for this purpose. Traditional methods in speech processing used statistical models, like Hidden Markov Models [Juang and Rabiner, 1991, Jelinek, 1997], to process the wave signal and recognize the words pronounced and understand the sentences. However, these methods were very limited in vocabulary and the complexity of the sentences that human users could use and the recognition rates were far from perfect. Today, with the advent of GPUs, as in the previous sections, deep learning techniques are becoming the focus of interest of researchers. End-to-end architectures, like that proposed in [Amodei et al., 2016, Xiong et al., 2018, Chan et al., 2016], mainly based on a combination of CNNs, for extraction of features, and RNNs, for temporal information analysis, are taking the lead and obtaining interesting results.

In the case of social robotics, speech recognition is an important fea-

ture, as we need an intuitive, organic and more natural method of communication than the old-fashioned peripherals. In [Yang et al., 2018], the researchers propose the use of the Nao robot to maintain conversations with children with autism and automatically extract crucial information on their interests to recommend them picture books. In [Mavadati et al., 2016], the authors propose a conversational therapy using a Nao robot that encourages the child to talk about their experiences and help them to recognize objects and imitate facial expressions, as shown in Figure 5.9. As a different approach, in [Yabuki and Sumi, 2018], the authors use a Pepper robot to teach people with typical development to communicate with people with autism spectrum disorder.



Figure 5.9: Conversational robot implemented with Nao to help autistic children to recognize objects and imitate facial expressions . Extracted from [Mavadati et al., 2016].

All of these studies show that not only patients with autism can benefit from the advent of the SAR and AI techniques, but therapists and family members also have more tools to help them with therapy and day-to-day living.

5.2.4 Conclusions

Socioeconomic changes and the lack of healthcare professionals to cover the unceasing demand of services and care have led to the need for technological solutions to mitigate this situation. In addition to intelligently interacting with the environment, the techniques developed must be suc-

cessfully adopted by users. In this sense, neuroscientific evidence shows that users, especially children, tend to engage with robots better than traditional screens and their design must make the user feel comfortable and increase their well-being. As a consequence, the scientific response to these issues is AR and, more precisely, SAR, which integrates a human-robot interaction in a social way.

This section presents an overview of the state-of-the-art SAR solutions for helping and assisting older adults in their daily activities such as activity scheduling and rehabilitation; and for helping children with autism spectrum disorders by means of diagnosis and social therapies. These solutions benefit from new advances in AI, as these increase the autonomy levels of assistance robots, allowing them to adapt to unforeseen circumstances without the direct intervention of a human. Thus, the advent of SAR along with AI can help users with their day-to-day living, promoting their daily functioning, well-being and independence.

Despite the active development in social assistive technology, there is still work to be done. Indeed, the current solutions do not provide ideal solutions to all needs of people with disabilities, but the results are highly promising.

5.3 EVA: Evaluating at-home rehabilitation exercises using augmented reality and low-cost sensors

In this section, we propose an augmented reality system, EVA, to perform and evaluate rehabilitation exercises at home. This system is aimed at two kind of people: patients who require rehabilitation at home after an injury, and the elderly people. Therefore, our purpose is to help them to recover from their affections and, consequently, to improve their quality of life.

Our proposal consists of spawning a personal trainer on the patient's home by taking advantage of the augmented reality methods. The user is able to watch the personal trainer and carry out the exercises by imitating him in real time, just like if they were in an actual gym. Upon the end of an exercise, the system automatically grades the patient's performance taking into account the similarity between the trainer's movements and the patient's ones. The exercise sessions are recorded such that the patient and the therapist could review them anytime in order to improve their performance and know the patient's health status at any time. In addition, low-cost sensors like regular colour cameras are used, making the system easily affordable.

We conceived the approach as a cloud-based service. The heavy computation part of the system is carried out in remote servers that are maintained by the entity that offers the service. Namely, the government, hospitals, clinics or retirement homes for instance. The final user only needs a low-end terminal like an embedded device or a tablet/smartphone. Despite the computation power requirements are high in the server side, a single machine could render service to several clients.

Summarizing, the main contributions of this section are:

- A low-cost AR rehabilitation app which successfully integrates different deep learning methods.
- Remote rehabilitation, making therapy accessible to users who would otherwise not have access. This feature is desirable as stated by the

WHO [World Health Organization, 2015].

- Non-intrusive. It relies only on vision algorithms.
- The integration of a *reference mat*, who shows how to do any required movement. This is crucial to guide the patient when at-home rehabilitation takes place and the therapist is not present

5.3.1 Proposal

In this section, we propose an augmented reality system to perform and evaluate at-home rehabilitation exercises. This system is aimed at two kinds of people: patients requiring at-home rehabilitation after an injury, and elderly citizens. Thus, our purpose is aiding them to recover the mobility of the affected limbs and, consequently, to improve their quality of life.

With that aim, our system comprehends several rehabilitation exercises that are displayed in a television, computer screen or projector. These exercises are performed by a virtual personal trainer that appears in the patient's room through augmented reality. So, the display acts as the mirror in a gym: the user is watching himself aside the virtual personal trainer. Then, the user must choose an exercise and mimic the virtual trainer movements. After each exercise, a score is given so that the patient has immediate feedback about their performance.

In rough lines, the system's workflow can be summarized as follows (see Figure 5.10): when an exercise is running, the system first takes an image of the scene using a regular camera. Then, it looks for "*the trainer's mat*". This is an object that is used to estimate the floor plane and to set a common 3D reference frame. Once the floor plane is detected, a person detector is then used to extract the person's position within the image. The Area Of Interest (AOI) corresponding to the person is sent to the human 3D pose estimator that returns the 3D pose of the person. After that, the virtual trainer and the user interface are rendered over the image to be properly displayed on the screen. When the exercise is completed, the stored 3D human poses and the virtual trainer's poses are used to

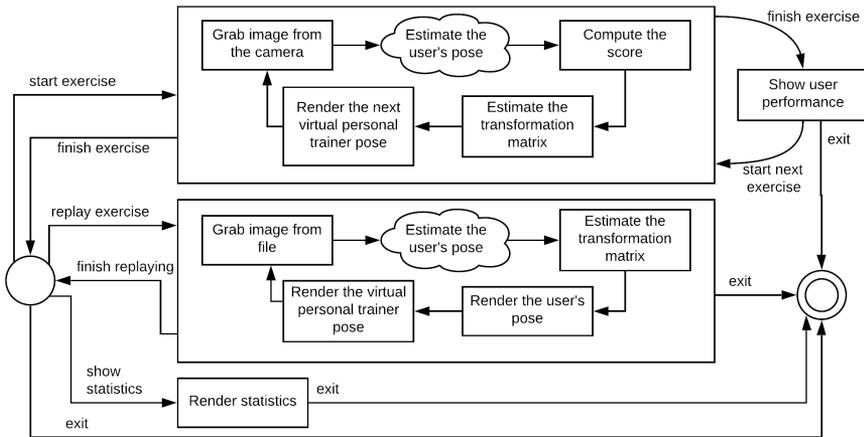


Figure 5.10: Flowchart of EVA system. Note that the cloud-based parts are shown in a cloud shape.

measure the user’s performance. This measure is displayed to the user together with their feedback, and is also stored for further analysis.

It is worth noting that the virtual trainer performing the exercises has been recorded beforehand and the corresponding 3D poses were also estimated offline. The estimation of the trainer’s poses is computed following the same method we used for the user’s ones. In this way, the system is only rendering the trainer’s poses like a video.

As above mentioned, each exercise execution is stored allowing the user or the therapists to replay their exercises to spot mistakes and further improve their performance. By taking advantage of augmented reality, the user can show the replays in the room’s floor or even in their desk. Furthermore, our system EVA features a web service that accepts an image stream and renders on it the replay of an exercise session. The web server follows the client-server methodology to provide remote access to the capabilities of the system. This option could be used to build a smartphone and/or tablet application to show the replays anywhere and anytime by connecting with the system via WiFi or even 4G. Note that the scores are also stored with the purpose to provide user’s statistics to both users and therapists.

Additionally, EVA has an easy-to-use, friendly user interface based on

virtual touch buttons leveraging augmented reality.

5.3.1.1 Common 3D coordinate frame estimation

As above mentioned, the common 3D coordinate frame is estimated by detecting "*the trainer's mat*". For that, a well-known chessboard pattern has been used (see Figure 5.11). It must be placed on the floor away from the user. In this way, the common 3D coordinate frame allows EVA to accurately detect the floor plane and properly render the avatar of the virtual personal trainer in the scene. As a result, the user senses the virtual trainer next to them in the room.



Figure 5.11: A chessboard pattern is used as "*the trainer's mat*". It is a well-known pattern that allows the system to estimate a common 3D coordinate frame. The leftmost image depicts the pattern with the 3D axis superimposed. The rightmost image depicts the personal trainer (skeleton in green) in augmented reality in front of the patient who is imitating them. The personal trainer is rendered *in situ* thanks to the pattern's detection.

Although some state-of-the-art approaches are able to detect the floor plane without placing any pattern within a scene (e.g. [Yang et al., 2016, Mur-Artal and Tardos, 2017]), their lack of accuracy, the high complexity of their calibration procedure or the need of non-static cameras make them unsuitable for the problem at hand. In addition, the presence of a physical element in the room prevents the user to inadvertently trespass the trainer *personal* space, what would negatively affect the perception of the augmented reality.

With the aim to detect the floor plane and to set a common coordinate frame, EVA firstly looks for the intersections between the pattern squares

in the image. As a result, a set of 2D points on the image plane is obtained. Then, as the pattern is known, the corresponding 3D points can be easily stated. The points lay in a plane so the Z-value is 0. Then, the length of each square side is also known. Therefore, we can easily compute a set of corresponding 3D points for each 2D point detected in the color image. Finally, we solve the Perspective-n-Point problem. As a result, a transformation matrix is obtained. This matrix transforms a point from the 3D world coordinate frame to the 2D coordinate frame of the image.

Finally, it is worth noting that the intrinsic camera parameters are required to solve the Perspective-n-Point problem and, as a consequence, the camera must be calibrated. This calibration process is carried out offline.

As the personal trainer poses are stated in the 3D space, they can be easily translated into the image coordinate frame from the transformation matrix computed in this stage. This helps to achieve the illusion of the personal trainer being actually aside the user in their own living room. Furthermore, the trainer is scaled according to the user's size. This virtual scaled view is a useful feature since it adjusts the taken video recordings to the chessboard pattern size. So, for instance, the therapist could reproduce any exercise session by means of a reduced chessboard pattern on its desk.

5.3.1.2 Human 3D pose estimation from monocular frames

With the aim to compare the user's and the trainer's movements in a robust way, a 3D pose is mandatory. A solution could be the estimation of the 2D human poses since low processing load techniques can be found in the literature [Cao et al., 2017]. However, the 2D poses may cause ambiguities and singularities between two poses. This fact could lead to a bad similarity estimation. This is the case depicted in Figure 5.12, where it is impossible to state whether the user is rising his arm forward or backward.

Due to the required accuracy in human pose comparison, a 3D pose estimation approach has been used. Our 3D human pose estimation system is based on the Human Mesh Recovery (HMR) approach [Kanazawa et al., 2018]. This method consists of an end-to-end framework for reconstruct-

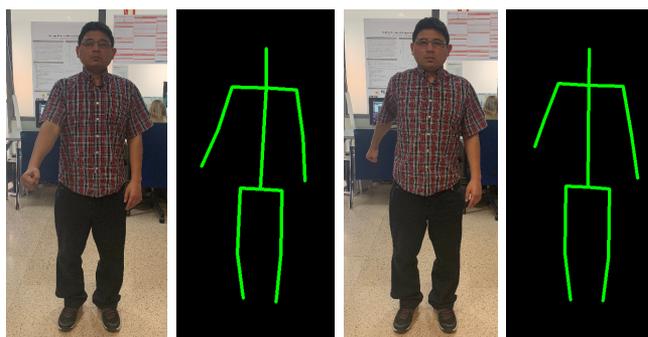


Figure 5.12: The 2D poses cannot be used for a robust pose comparison between the user and the trainer because it may eventually fall into a singularity or ambiguous poses such as the depicted in this figure. According to the 2D pose, it is impossible to state whether the patient is rising his arm forward or backward.

ing a full 3D mesh of a human body from a monocular *RGB* image. So, a deep learning-based encoder is able to predict the camera pose, the person's shape and the person's pose for each taken image. These predictions are used to render a model which is then validated by a discriminator. The discriminator is able to state if the predictions correspond to a real person or not. Given that this approach does not use a 2D intermediate representation being able to make final predictions in one forward step, it is very fast. Note that EVA only works on the inferred 3D human pose, discarding the camera pose and the person's shape. This 3D human pose is expressed as a list of 3D points in camera coordinates, corresponding to 19 joints of the human body.

This approach for 3D human pose estimation is able to accurately estimate the human pose even under different scenarios as illustrated in Figure 5.13. For instance, the system works well for persons sitting in wheelchairs. It can also deal with high levels of self-occlusion. These cases play a main role when working with physical injured people and elderly.

Nonetheless, this approach has an important issue to be pointed out. The best human pose estimations are obtained when there is only one person in the image. However, our system captures images that includes the trainer's mat and the room elements apart from the patient. For that reason, a person detection stage is needed. To carry out this process, we leveraged YOLOv3 [Redmon and Farhadi, 2018]. It is a well-known

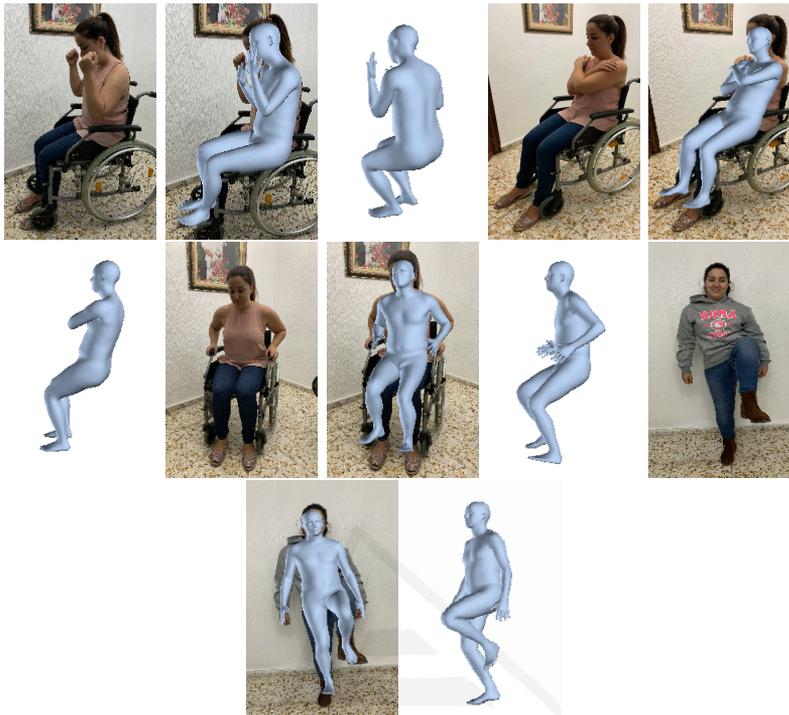


Figure 5.13: . The 3D human pose estimation system is able to accurately estimate the human pose even in difficult cases such as persons in wheelchair or scenarios with high self-occlusion level caused by the exercises.

method for object detection and recognition, providing a decent accuracy while keeping the computation cost at bay. This region CNN architecture is able to detect the position of the objects in the image plane, estimate the label of those objects as well as their corresponding confidence score. This architecture achieved a 0.51 mAP (measured over the intersection over union) on the test set of the COCO MS dataset [Lin et al., 2014b]. From this pretrained model, a wide range of objects including persons, can be detected. Figure 5.14 shows the performance of this architecture for person detection.

So, the complete human 3D pose estimation pipeline can be described as follows: first, an image is grabbed from the camera. Then, this image feeds the YOLOv3 detector, which provides the AOI of several objects. From them, the patient is identified as the AOI with the best confidence score for person. This AOI is cropped from the original image and for-



Figure 5.14: The person detector YOLOv3 [Redmon and Farhadi, 2018] is able to accurately estimate the AOI enclosing the patient.

warded to the HMR network. As a result, the 3D pose of the patient is obtained. Figure 5.15 shows the performance of the HMR network feeding it on the whole image and on the person area detected by YOLOv3. As it can be observed, the predictions are more precise when the network is fed with just the person area. It is worth mentioning that we use the models released by the original authors from both HMR and YOLOv3 methods.



Figure 5.15: Performance of the Human Mesh Recovery network feeding it the whole image (center) and the person AOI detected by YOLOv3 (rightmost).

5.3.1.3 Scoring the user's performance

With the purpose to evaluate the user's performance during rehabilitation exercises, we propose to compare the user's joints trajectory with the trainer's ones by using a Dynamic Time Warping (DTW) approach [Berndt and Clifford, 1994]. This state-of-the-art method is aimed to find patterns in time series. So, it finds a warping path to align the elements of two

time sequences such that the distance between them is minimized. This distance between two elements must be defined, being the euclidean distance the most common. We can see the DTW problem as a minimization of the cumulative distance over the whole possible paths of two time series elements.

This method provides a score quantifying the degree of adjustment of two times series when stretching or compressing their elements along the time. Note that this score stays in the range $[0, 1]$ when comparing different series. For that, the distance measurement is often modified to be relative to a baseline distance.

Given that the time dimension is not considered, this method is appropriate to compare two 3D human poses separated by a short period of time as it is the case (i.e. the user may take time to imitate the trainer's movements due to their disability).



Figure 5.16: Joint positions considered by the system as returned by the Human 3D Pose Estimation system and used to compare the human poses of the user and the virtual trainer.

Assuming that different patients use different cameras at different places, it is necessary to define a common reference frame to avoid the influence

of these factors in the comparison of the joint positions. We have to transform from the camera frame, defined by 3 camera axis (c) and the origin of coordinates (Q_0) in Equation 5.1, to a new invariant reference frame. This reference frame is defined using the user's body in a preset position. The reference point (P_0) of this frame is the neck (joint 1 in the Figure 5.16). Then, we build two perpendicular vectors (v_0 and v_1) from shoulder center to the head (joint 0) and to the right shoulder (joint 2) respectively. The last axis vector (v_2) is calculated applying the cross product of the previous vectors, as shown in Equation 5.2.

$$\begin{aligned}
 \text{Cameraframe} &: (c_0, c_1, c_2, Q_0) \\
 c_0 &= (0, 0, 1) \\
 c_1 &= (0, 1, 0) \\
 c_2 &= (1, 0, 0) \\
 Q_0 &= (0, 0, 0)
 \end{aligned} \tag{5.1}$$

$$\begin{aligned}
 \text{Bodyframe} &: (v_0, v_1, v_2, P_0) \\
 v_0 &= (j0.x - j1.x, j0.y - j1.y, j0.z - j1.z) \\
 v_1 &= (j2.x - j1.x, j2.y - j1.y, j2.z - j1.z) \\
 v_2 &= v_0 \times v_1 \\
 P_0 &= (j1.x, j1.y, j1.z)
 \end{aligned} \tag{5.2}$$

Once we have defined the reference frame centered in the body, we have to estimate the transformation matrix from the camera's frame to this reference frame. The transformation matrix is estimated from the Equation 5.3, that calculates the coefficients necessary to put the new basis vectors v_0 , v_1 and v_2 as a linear combination of the camera axis c_0 , c_1 and c_2 and the new origin of coordinates in terms of these vectors and the camera's origin of coordinates.

$$\begin{aligned}
 v_0 &= a_{11}c_0 + a_{12}c_1 + a_{13}c_2 \\
 v_1 &= a_{21}c_0 + a_{22}c_1 + a_{23}c_2 \\
 v_2 &= a_{31}c_0 + a_{32}c_1 + a_{33}c_2 \\
 P_0 &= a_{41}c_0 + a_{42}c_1 + a_{43}c_2 + Q_0
 \end{aligned} \tag{5.3}$$

$$M = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & 1 \end{bmatrix}$$

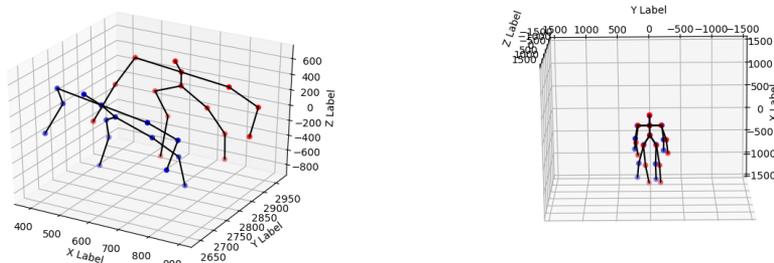
With the transformation matrix between the camera and the body reference frames, we can simply transform the points multiplying this matrix with the column vector of homogeneous coordinates of each point. Suppose that a and b are the homogeneous representation of the same point on the camera and the body reference frames respectively, so they satisfy the Equation 5.4.

$$\begin{aligned}
 a &= M^T b \\
 b &= (M^T)^{-1} a
 \end{aligned} \tag{5.4}$$

Then, we transform the whole points into the body reference frame and calculate the distance between two sequences of movements for every pair of joints using FastDTW, an accurate and efficient DTW implementation presented in [Salvador and Chan, 2007]. So, the final distance is the average of the accumulated distance of the joints.

Due to the fact that the final distance is estimated for every joint individually, we can report to the user which are the joints they are performing worse. In addition, the DTW is computed every 120 frames with an offset of 60 frames. In this way, we can even know which moment of the rehabilitation session has been performed accurately and which has not.

Figure 5.17 depicts a representation of joints in body coordinates for two different bodies. The left image shows the body joints without coordinate normalization, while the right image illustrates both body joints after coordinate normalization.



Joints without normalization.

Joints with normalization.

Figure 5.17: Representation of joints in body coordinates for two different bodies.

5.3.2 Augmented reality application

The browsing within the menus of the developed application is made through body motion. The user must lay the hand on a button to trigger the corresponding action. For that, the Human 3D Pose Estimation pipeline described in Section 5.3.1.2 has been used. So, the estimated 3D position of the right hand is translated into the image plane. If this point lays on a button, this button is selected (see Figure 5.18, where the user is selecting the *Replay* button in the EVA’s main menu). Note that the user must lay the hand on a button for 5 seconds to trigger the corresponding action. This prevents from inadvertently activating a button. In addition, the user interface distinguishing the two actions (i.e. selection and trigger) such that an arrow marks the selected button, whilst a progress circle in the bottom left shows the time left to trigger the corresponding action.

Additionally, a keyboard could be also used for browsing the menus. This feature is useful when the user does not want to perform a rehabilitation session but only replay it on their desktop, or review their statistics. In this way, it is not mandatory to set up the big “*trainer’s mat*” nor to use the body for browsing the application.

It is also worth noting that the user interface was created considering accessibility and usability. It features big and descriptive buttons with high contrast colors so that the use of the application is intuitive and



Figure 5.18: Screenshot of the main menu of the developed application. The user can select the different options from the menus with its own body by placing the hand behind the buttons during 5 seconds.

straightforward.

5.3.2.1 User interface

The first button of the main menu is “*Start*”. This option triggers a rehabilitation exercise session as depicted in Figure 5.19. So, after pushing the button, a list of exercises are shown where the user should choose the ones to be performed in the same way as before. Once an exercise session is selected, a virtual personal trainer is rendered over the “*trainer’s mat*”. The feed of the camera is being displayed so that the user is watching the trainer besides him in an augmented reality fashion. The user must imitate the movements of the virtual trainer upon the completion of the session. A plot in the left corner of the display is continuously showing the user’s update score. Finally, a summarized score representing the patient’s performance over the whole exercise is shown. In addition, a timeline of the session is also displayed so that the user can review which part of the session they did better and which one needs more work on.

The next button of the main menu is “*Replay*”. As above mentioned, this option allows both patient and therapist to replay an exercise session.

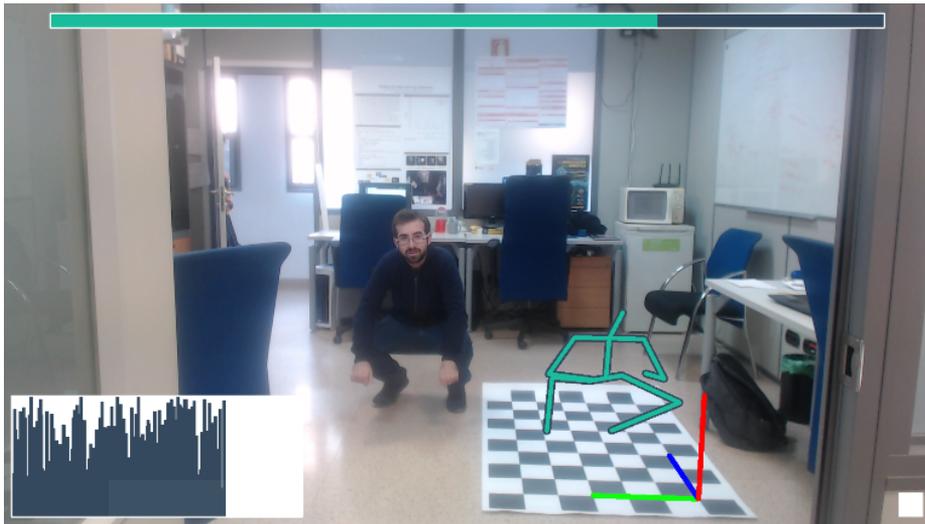


Figure 5.19: A sample of a patient performing a rehabilitation session. The virtual trainer is being rendered using augmented reality methods.

During the replay, the app shows the positions where the patient stood rendered in an augmented reality video-like way. The replay can be paused such that the therapist or the patient could rotate or zoom in/out a certain pose. This can be done by moving the "trainer's mat". The avatar is customizable so that it can render a person or a skeleton-like character. A score histogram is also displayed in the bottom left corner. This histogram shows the DTW score for each instant of the rehabilitation session. Finally, a progress bar in the top of the window shows how much of the replay is left. Figure 5.20 depicts a still of a replay being displayed. Note that, in this case, the therapist is using a reduced size "trainer's mat" allowing the app to render the augmented reality avatar right on their desk. The replay feature allows an easy analysis of the rehabilitation sessions with the goal of evaluating the patients' performance and modifying the rehabilitation program if needed.

The last entry of the main menu is "Statistics". This menu shows a new window with the history of the past sessions performed by the user. This window displays a summary of the scores and plots the patient's adherence and performance. These metrics could be used for the patient and the therapist for review and evaluation purposes.

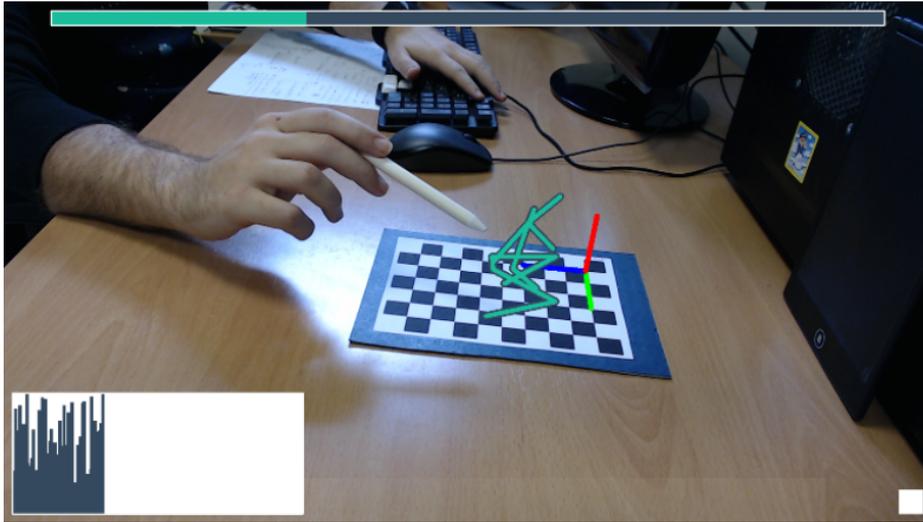


Figure 5.20: The image depicts a replay being displayed. Note that, in this case, the therapist is using a reduced size "trainer's mat" that allowed the application to render the augmented reality avatar right on his desk.

Finally, a very early version of a web service was implemented. This web service allows EVA to be offered as an external service. Therefore, patients can benefit from EVA wherever through low computation powered machines such as tablets, embedded computers and smartphones. In this way, the taken images from the user's terminal are sent to the remote server to be processed. So, the user can get their performance score in real-time independent of the computing power.

5.3.3 Rehabilitation exercises

An important step in any rehabilitation treatment is the home exercise program. This program consists in the patient's performance of prescribed physical exercises at home. In this sense, there is a wide variety of exercises depending on the body part to be recovered and/or their goal. In particular, a public exercise program suggested by the British National Health Security (NHS) to improve elderly's fitness and well-being has been considered in this chapter [British National Health Security (NHS), 2018]. This guide is composed of twenty-three exercises divided into four groups: flexibility, strength, balance and sitting. However, the therapist is who will

decide the exercise program to be followed by each patient. So, the virtual trainer reproduces the tailored set of exercises while analyzes the patient's evolution in their performance.

5.3.4 Experimentation and results

With the aim to validate EVA, two kinds of experiments were carried out. On the one hand, each subsystem has been individually evaluated in a qualitative and quantitative way. On the other hand, the whole system was qualitatively analyzed.

Firstly, the evaluation of the Human 3D Pose Estimation and the DTW Scoring subsystems were performed by using the KARD dataset, explained in Section 1.2.11.1 [Gaglio et al., 2015].

For these experiments the following hardware was used: an Intel Core i5-3570 with 8 GiB of Kingston HyperX 1600 MHz and CL10 DDR3 RAM on an Asus P8H77-M PRO motherboard (Intel H77 chipset). The system also included an Nvidia GTX1080Ti. The framework of choice was Keras 1.2.0 with Tensor Flow 1.8 as the backbone, running on Ubuntu 16.04. CUDA 9.0 and cuDNN v7.1 were used to accelerate the computations.

5.3.4.1 Human 3D pose estimation experiments

The accuracy of the Human 3D Pose Estimation was evaluated through the KARD. Thus, it was required to divide the activity videos in still frames. As described in Section 5.3.1.2, this process involves two different stages. Firstly, each video frame is forwarded to the YOLO architecture which provides the AOI of the patient within the scene. Then, this area is cropped and used to perform the 3D pose estimation with the HMR approach. As a result, the 3D position of the patient body joints is estimated. Then, the 3D points are projected to the image plane. Some results randomly chosen are shown in Figure 5.21. Note that the images were resized to 224×224 px since it is the architecture's input size.

Figure 5.22 illustrates the amount of samples per distance error threshold. Note that the amount of joints are expressed in percentages. As it can be seen, 92% of the joints yielded an error below 20 px, and 80% below



Figure 5.21: The top row depicts random samples extracted from the KARD and the bottom row shows the estimated human 3D poses.

12 px. The mean error averaged across the entire dataset is 9.58 px for an image of 224×224 px resolution. As the results show, this approach is accurate enough to be used for a successful human pose detection from a single RGB frame.

In addition, we also benchmarked the Human 3D Pose Estimation system in the 3D space using the KARD. In this case, we also use YOLO to detect the person in the scene and then forward only the AOI to the pose estimator. As the 3D pose returned by the system yields no scale, the tridimensional positions of the joints are arbitrary located in the space, yet depicting the correct pose. To enable a fair comparison, we followed the method described in Section 5.3.1.3. Thus, we express all the poses in a coordinate frame local to the body pose. Then, as the scale is also different, we computed a scale factor using the norm of the vector that goes from the neck to the right shoulder. This process is applied to both ground truth and estimated poses. As a result, both poses are in the same coordinate frame and yield the same scale. Finally, we measure the mean euclidean distance between each joint averaged across all the frames.

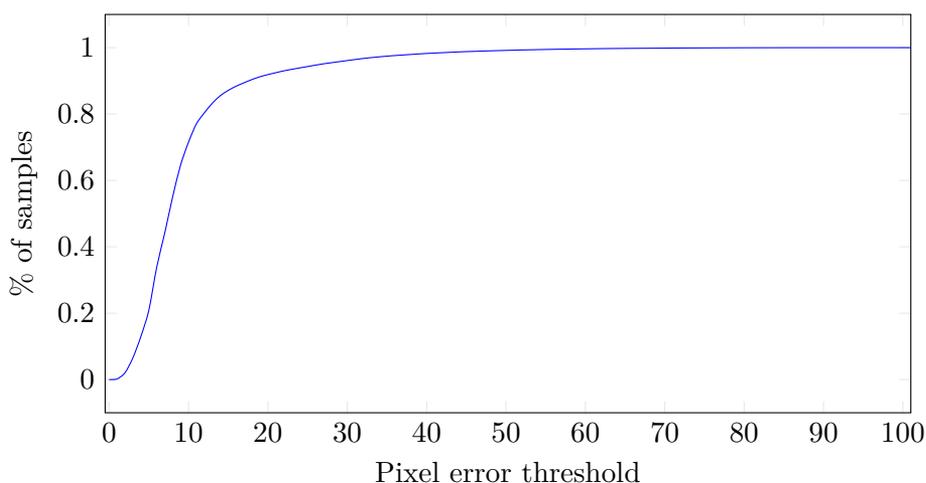


Figure 5.22: The accumulated percentage of samples per error (px on a 224×224 image) threshold between the 2D points provided by KARD and those estimated by our Human 3D Pose Estimation approach.

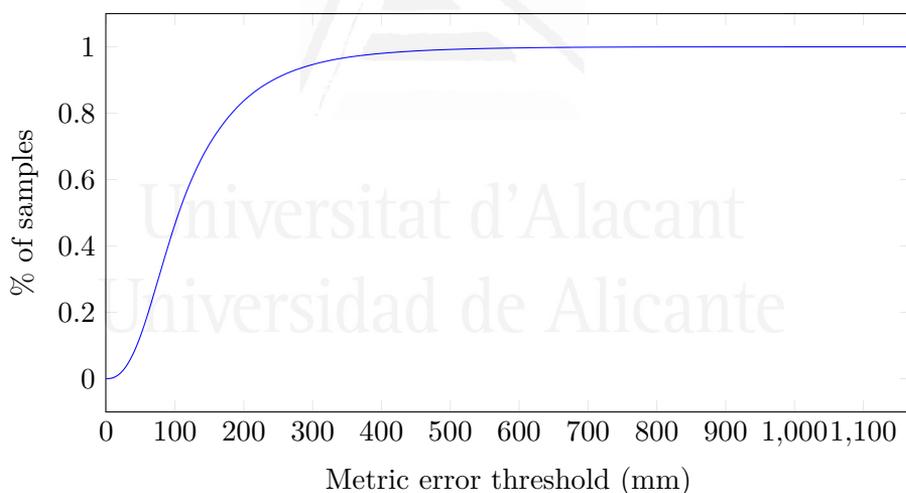


Figure 5.23: The accumulated percentage of samples per error (mm) threshold between the 3D points provided by KARD and those estimated by our Human 3D Pose Estimation approach.

As shown in Figure 5.23, the 70.66% of the samples are under the 150mm threshold. Note that this threshold is the same as that used by the authors of the HMR system to report the percentage of correct keypoints. In their proposal, they achieved a 72.9% on the MMPI-INF-3DHP dataset.

In addition, we have obtained a mean error of $129.52mm$.

These experiments demonstrate the accuracy of the system. Regarding the 2D pose estimation, it can be concluded that it is accurate enough to provide a good representation on the screen, what is important to encourage both users and therapists to use the application. Nonetheless, the 3D poses computed by the system are approximated. Therefore, the obtained evaluation is a reference, but not a medical diagnosis.

5.3.4.2 DTW scoring experiments

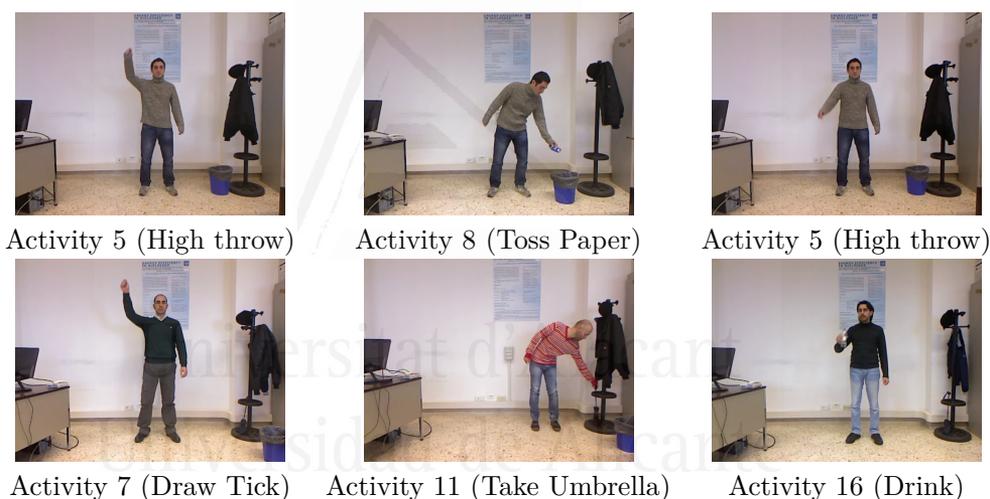
The KARD has been also used to evaluate the effectiveness of the scoring system. As above mentioned, this dataset contains 18 activities and each activity is performed by 10 different subjects. We compare them to each other. The effectiveness of the DTW is determined in the following way: the lower the value obtained, the closer or similar the activity will be. Moreover, the different score will be then compared against another activity. The obtained results are shown in Table 5.3 where each value indicates the similarity between the activity in the row with respect to that in the column. So, the value in bold represents the closest similarity, while a higher value indicates a higher difference between two activities. As it can be observed, the system is clearly confused between the activity pairs 7 – 5, 11 – 8 and 16 – 5. This is due to two main factors: (1) an overlap of movements (7 – 5, 16 – 5), and (2) the similarity in the activity movements (an object is released in activity 8 whilst an object is grabbed in activity 11).

5.3.5 Conclusions and future work

In the face of the relentless demand for rehabilitation services and the lack of resources to access them, we propose EVA, an augmented reality application for evaluating rehabilitation programs at home. This low-cost application only requires a regular camera to capture and evaluate the 3D patient's pose. So, EVA stores the rehabilitation sessions such that patients and therapists could review them and adjust the exercises accordingly. EVA was qualitative and quantitative evaluated to show their suitability.

Table 5.3: Confusion matrix for the DTW Scoring experiments with the KARD.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15	G16	G17	G18
G1	0,19	0,29	0,43	0,28	0,22	0,36	0,32	0,31	0,24	0,23	0,47	0,35	0,26	0,81	0,66	0,32	0,42	0,34
G2	0,29	0,15	0,31	0,25	0,19	0,36	0,29	0,29	0,22	0,22	0,45	0,32	0,25	0,78	0,61	0,33	0,39	0,3
G3	0,43	0,31	0,22	0,34	0,32	0,46	0,41	0,36	0,32	0,35	0,52	0,42	0,36	0,9	0,73	0,46	0,53	0,44
G4	0,28	0,25	0,34	0,12	0,17	0,34	0,29	0,23	0,15	0,19	0,41	0,28	0,19	0,78	0,65	0,34	0,34	0,22
G5	0,22	0,19	0,32	0,17	0,01	0,29	0,21	0,18	0,08	0,09	0,4	0,21	0,11	0,79	0,6	0,25	0,3	0,17
G6	0,36	0,36	0,46	0,34	0,29	0,26	0,35	0,35	0,29	0,32	0,48	0,41	0,32	0,79	0,65	0,41	0,49	0,4
G7	0,32	0,29	0,41	0,29	0,21	0,35	0,23	0,31	0,23	0,24	0,48	0,35	0,26	0,8	0,65	0,36	0,42	0,32
G8	0,31	0,29	0,36	0,23	0,18	0,35	0,31	0,07	0,14	0,19	0,29	0,2	0,19	0,76	0,72	0,39	0,28	0,18
G9	0,24	0,22	0,32	0,15	0,08	0,29	0,23	0,14	0	0,1	0,36	0,19	0,13	0,75	0,65	0,3	0,27	0,13
G10	0,23	0,22	0,35	0,19	0,09	0,32	0,24	0,19	0,1	0,02	0,4	0,22	0,14	0,77	0,64	0,3	0,29	0,17
G11	0,47	0,45	0,52	0,41	0,4	0,48	0,48	0,29	0,36	0,4	0,31	0,4	0,42	0,82	0,79	0,54	0,43	0,38
G12	0,35	0,32	0,42	0,28	0,21	0,41	0,35	0,2	0,19	0,22	0,4	0,13	0,21	0,77	0,74	0,42	0,29	0,2
G13	0,26	1,09	0,36	0,19	0,11	0,32	0,26	0,19	0,13	0,14	0,42	0,21	0,05	0,77	0,65	0,32	0,31	0,19
G14	0,81	0,78	0,9	0,78	0,79	0,79	0,8	0,76	0,75	0,77	0,82	0,77	0,77	0,75	1	0,83	0,83	0,78
G15	0,66	0,61	0,73	0,65	0,6	0,65	0,65	0,72	0,65	0,64	0,79	0,74	0,65	1	0,6	0,65	0,76	0,69
G16	0,32	0,33	0,46	0,34	0,25	0,41	0,36	0,39	0,3	0,3	0,54	0,42	0,32	0,83	0,65	0,26	0,46	0,37
G17	0,42	0,39	0,53	0,34	0,3	0,49	0,42	0,28	0,27	0,29	0,43	0,29	0,31	0,83	0,76	0,46	0,2	0,23
G18	0,34	0,3	0,44	0,22	0,17	0,4	0,32	0,18	0,13	0,17	0,38	0,2	0,19	0,78	0,69	0,37	0,23	0,02

**Figure 5.24:** The first row shows some sample activities from a certain class and the second row depicts different samples from different classes that causes confusion in the system. As the images depict, the confusion is expectable as the classes are very similar despite being labeled as different.

For that, two different types of experiments were carried out. Firstly, each integrating module was individually tested and then a complete evaluation was performed.

Despite the promising results, the experiments brought to light an EVA limitation in terms of human pose estimation. The human pose estimation sometimes fails when people has an amputated limb since it assumes that

the images depict a canonical person and it tries to estimate the position of all their joints. This fact will be pointed out in the future since a considerable amount of EVA's users may present this handicap.

Apart from that, we plan to further explore the web service potential and the idea of a smartphone application. It would not be only able to show replays, but also to provide statistics to the therapists and patients and act as a client as well. We also plan to extend the amount of exercises and to add more customization features to our proposal. For instance, more trainer avatars or new user interface themes to help the visually impaired, will be added. The possibility to add wearable sensors like smartbands will be also explored in order to gather more information about the user's health status (e.g. ECG or heart rate). In this way, the therapist could have more information to properly adapt the rehabilitation process. Finally, a native VR/AR environment like Vuforia [PTC, 2019] would be used to leverage and enhance the visualization part.

5.4 Semantic visual recognition in a cognitive architecture for social robots

The learning process of robots is still far from the learning process of humans. Recent advances in CNN have revolutionized the ability of machines to visually recognize classes of elements in images. Despite its high effectiveness, the learning process is offline, slow, and requires much computation. Humans, in contrast, learn by accumulating knowledge from experience.

Distinguishing a person is a desirable capacity of a social robot. Robots meet different people during their operation. In many tasks, they are required to distinguish them from others. Tasks such as serving drinks in a bar, following a person in a crowd, greeting the people around them by name, among others, are commons.

This section presents the results of an investigation that mixes DL techniques and statistical classifiers to achieve a robust system of online learning and recognition of people. The proposed approach has different operating modes. In the learning mode, the robot learns to distinguish the person with whom it is interacting. The robot extracts the person's name from its interaction with the person. The aim is that, after several seconds of communication, the robot will be able to distinguish it from the rest of the people. In normal mode, the robot can identify all the people from whom it has learned to carry out tasks entrusted to it. This approach is not only valid for people, but can also be used to identify any visual elements.

The proposed approach has several advantages over current methods. The most obvious is that it does not detect classes of elements, but distinguishes particular instances within the same category. While works like [Krause et al., 2013] identify instances of a given object, we, however, are able to track those instances over time.

The second advantage is its operation mode. Current DL approaches would require an offline process of labelling, training, and deployment on the robot. The proposed approach can alternate training phases with operation phases without turning off the robot. Moreover, this learned

knowledge between robot operations is preserved. This technique is similar to online learning methods such as [Charalampous and Gasteratos, 2016] and [Bae and Yoon, 2014], as the model can be adapted to new knowledge received in execution time. Last but not least, learning a person requires less than 15 seconds of interaction (this is the maximum time that some competitions establish, as explained in the next paragraph). Such training times are unimaginable in current methods that use only DL.

Specifically, the main contributions of this chapter are:

- Person Identification Memory System (PIMS).
- An exhaustive experimentation of different classifiers focused on processing time and accuracy for the PIMS.

The proposed approach combines DL architectures trained with a new dataset with feature extraction techniques. Some previous works also created a framework to combine different visual features [Li et al., 2018b] with a similar approach to the proposed one, but this approach uses DL embeddings instead of handcrafted features.

5.4.1 Person identification memory system

The goal of the PIMS module is to identify people rapidly and accurately, and learn and memorize new subjects on the fly. The techniques used to perform this task are a combination of DL architectures with traditional classifiers.

The DL architectures are used in order to generate the embeddings and features to recognize people, which perform much better than traditional approaches. However, they require a lot of time in the training stage, so they are trained offline and their weights will remain frozen on the live learning stage. In the case of traditional classifiers, their accuracy is poorer compared with the previous ones, but they require little time to be trained. Consequently, they can be retrained live and adjust their parameters to the new recognition requirements. The combination of both methods leverages their strengths and offsets their weaknesses.

The architecture of the proposal is shown in Figure 5.25. In the training stage the system receives images of the subject to be learnt, which

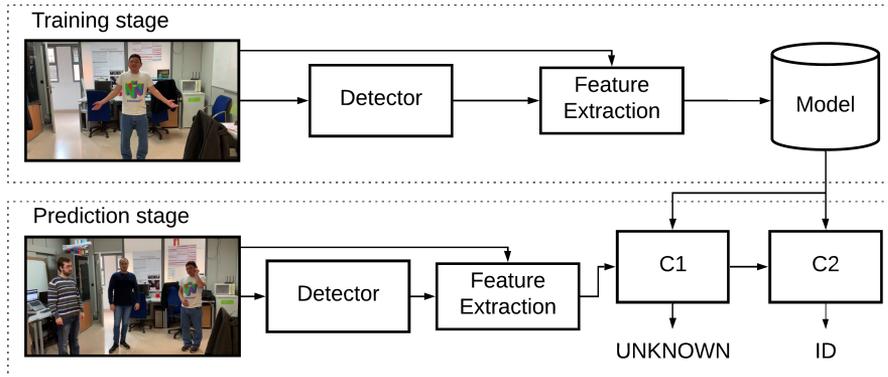


Figure 5.25: Diagram of the proposal. At the training stage, the detector calculates the bounding box around the person, a DL network extracts their features and the vector representation is sent to the classifier’s model, which is created by gathering labelled data. At the prediction stage, the detector and the feature extractor work as on the training stage, the C1 classifier rejects the subjects that are unknown and the C2 classifier distinguishes people’s identities.

moves in front of the camera with different postures. For every frame, the Detector locates the person with a bounding box inside the image. The Detector consists of a RCNN capable of predicting the location of subjects in an image and returns the AOI of every person. In this case, the architecture used is YOLO v3 [Redmon and Farhadi, 2018]. Subsequently, the AOI is served as input to a modified Resnet50 [He et al., 2015]. This architecture is a state-of-the-art CNN for classification tasks. In order to take advantage of the generated features, the fully connected layer at the end of the network, which is used for classification tasks, has been removed, so this network acts as a feature extractor. Therefore, the output of the network is a feature vector of 2048 values which is labelled with the person ID and sent to the classifier’s model to be learnt. The label is known as this is performed at training time. Although YOLO extracted features could be processed directly, instead of applying another network’s output, previous experiments show that using Resnet as feature extractor outperforms YOLO.

In the prediction stage, for every frame the Detector calculates the location of the people in the scene and generates the AOI. These AOI are

then sent to the Feature Extractor, which calculates a vector representation for every subject. These features are forwarded to the C1 classifier, which distinguishes between known and unknown people. Finally, if the person has been classified as known, their features are sent to the C2 classifier, which recognizes their identity.

The need for a separate classifier that performs the differentiation between known and unknown people is a result of the difficulties in setting a proper distance threshold inside a class-instance classifier for this purpose. In the literature, there exists a family of methods, known as anomaly detectors [Chandola et al., 2009], that carry out the differentiation between "normal" and "abnormal" data (in this case, normal data is known persons and abnormal data is unknown persons). In the semi-supervised case, they are capable of building their models with only "normal" data, which is perfectly suitable for this problem, as there is only data of known people. The hyperparameters of these classifiers can be optimized via automatic hyperparameter optimization such as random or Grid Search (GS), both used in the machine learning suite *Scikit-learn* [Komer et al., 2014].

With this previous step, the final class-instance classifier will always receive a known subject and can perform the classification without applying any kind of filtering threshold. If the received person is unknown, they will be rejected in the first step.

As a result of the combination of DL and traditional classifiers, the PIMS approach trains rapidly as the training samples are inserted in traditional classifier models and there is no need to retrain the DL models which calculate the features. Moreover, its accuracy is high as it takes advantage of DL architectures for detection and feature extraction. Finally, the system can also learn new classes (unforeseen person IDs) without any architectural modifications. In contrast, a pure DL approach would require modifications on the last layer and a retraining process.

5.4.2 Experimentation

In this section, the experimentation carried out to evaluate and validate the proposed approach is described. In addition, the details of the dataset used in the experiments are also reported.

The experiments were carried out using the following setup: Intel Core i5-3570 with 16 GiB of Kingston HyperX 1600 MHz and CL10 DDR3 RAM on an Asus P8H77-M PRO motherboard (Intel H77 chipset). The system also included an Nvidia GTX1080Ti, which was used for DL model inference. The framework of choice was Keras 1.2.0 with Tensor Flow 1.8 as the backend, running on Ubuntu 16.04. CUDA 9.0 and cuDNN v7.1 were also used to accelerate the computations. All the reported time measurements were made on this hardware.

5.4.2.1 Dataset

A custom dataset was recorded in order to test the proposed approach. This dataset was divided into two sets: training and test videos. The training set involved individuals standing in front of the camera and turning 360 degrees for 10 seconds. The test set consisted of different videos where the previous individuals moved around the scene freely for 20 seconds. Finally, an additional video was recorded with three of the subjects walking around the room and interacting with one another. The last video was used for qualitative evaluation and the rest for quantitative benchmarking. The total size of the dataset was 9 videos, recorded by a 12 MPx color camera at 1080p resolution and 30 fps.

In the experiments described in the following sections, all four training videos were used to build the recognition models. Then, these models were used to perform inference on the test videos. As the test videos only showed one person, the system performance could be evaluated directly.

5.4.2.2 RoboCup challenge

The proposed problem to be solved was *Carry my Luggage [Party host]* from the Robocup@Home 2019 competition [Robocup, 2019]. In this challenge, the robot must help the operator to carry some luggage outdoors.

First, the target person stands in front of the robot. The robot can give orders to move (turn round, move closer...) and takes pictures to recognize the person at that moment. Once the learning stage has concluded (20 seconds as maximum), the operator turns around and starts walking

in different directions and crossing paths with other people. The robot must be able to follow its target even with occlusions and unknown people around the environment. Figure 5.26 shows a recreation of the challenge captured from a Pepper Robot.

5.4.2.3 Person identification system experiments

In this set of experiments, full body person identification was benchmarked. The person detector, which was based on YOLO, ran a model trained on the COCO MS [Lin et al., 2014a] dataset as provided by the original author. This model was able to predict AOI of different objects but as people were the subject of interest, the other predictions were ignored. In addition to the detector network, the ResNet50 trained on ImageNet was used as the AOI feature extractor. VGG16 [Simonyan and Zisserman, 2014] and MobileNet V2 [Sandler et al., 2018] were also tested as feature extractors. The ResNet50, VGG16 and MobileNet were trained on the ImageNet [Deng et al., 2009] dataset as provided by the Keras framework.

For the first experiment, the goal was to identify the best classifier that distinguishes between known people instances. The performance of different classifiers was tested using the features obtained by the previous network, considering that every sample was known by the system. The training video consisted of the individuals completely showing themselves. By sampling them at a fixed frame skipping value, the model was able to capture the features for each pose, thus leading to high accuracy rates and reducing the processing time. The proposed approach was tested ranging the frame skipping parameter from 0 (all frames are used) to 20 (around 65 frames remained from each training video). The classifiers trained are KNN, whose backbone yields 10 trees. At inference time, only the nearest neighbor is used; SVM with radial basis function and linear functions; and Random Forest (RF). As setting the correct parameters directly impacts the accuracy, C and γ for the SVM were automatically computed by using GS. The values we used were $C = [0.1, 1, 10, 100]$, $\gamma = [1, 0.1, 0.01, 0.001, 0.00001, 10]$. The number of trees for the RF classifier were also automatically computed by GS. In this case, $n = [20, 50, 100, 200, 500, 1000]$. The depth of each tree

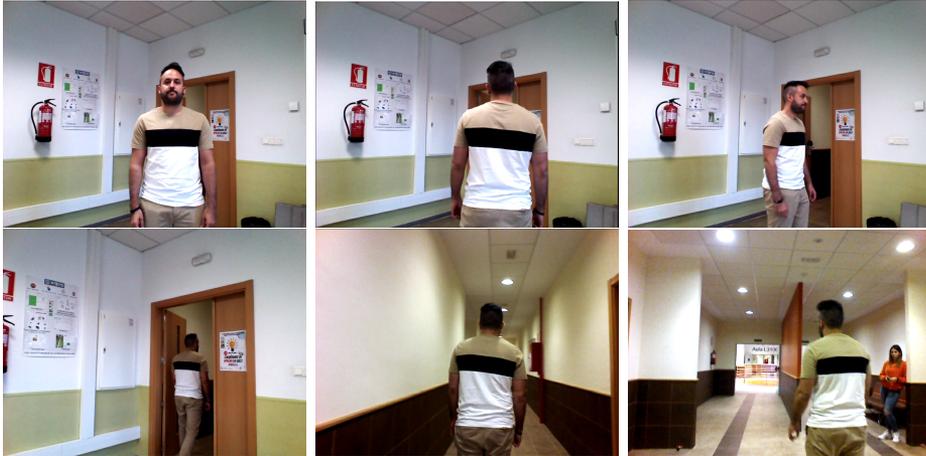


Figure 5.26: Different moments of a recreation for the proposed challenge. The first 3 images correspond to the training stage and the last 3 correspond to the test stage.

is computed by expanding all nodes until all leaves are pure or until all leaves contain less than 2 samples. Finally, Naive Bayes (NB) and Decision Tree (DT) were also involved as classifiers. These algorithms have no configurable parameters. We applied GS when necessary because the parameters are dependent of the data, and different datasets could require different parameters to perform properly. Manually setting these parameters is not feasible because the robot should be able to set them in an unattended fashion. All the methods were set to multiclass classification. The accuracy rates and F1-scores are as shown in Figures 5.27, 5.28, 5.29, 5.30, 5.31 and 5.32.

Regarding our dataset, as depicted in Figure 5.33, the SVM classifier outperformed the rest and the accuracy was maintained around 93% regardless of the frame skipping parameter. However, KNN and RF performed well, with an average precision of 90%. Despite the model containing fewer samples as the frame skipping increased, the accuracy was sufficiently high. This is because the model has enough semantic information in every case. It is worth noting that the model generated for frame skipping equal to 20 only contained 65 samples, around 16 for each individual. Random samples with the predictions superimposed are shown in Figure 5.34 for qualitative evaluation.

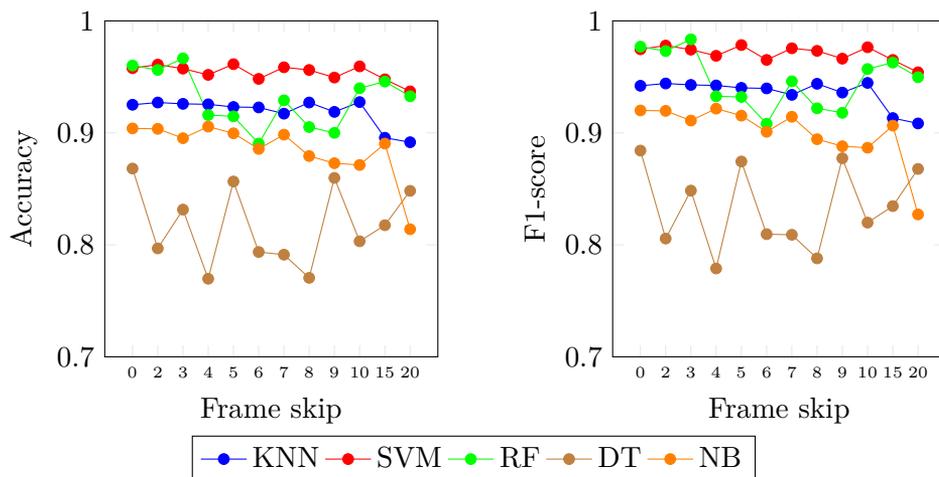


Figure 5.27: Accuracy and F1-score results for OUR's testing videos using ResNet50 with frame skip from 0 to 20.

In addition, the proposal was tested with the KARD, explained in Section 1.2.11.1. Despite this dataset being intended for action recognition tasks, it could be used to test the proposed approach as it had each person labelled independently. As there were a vast number of videos, only 5% of them were taken for training. The number of training frames was thus approximately the same as in the last experiment for each frameskip parameter, and so the results can be compared. The remaining 95% of the videos were used for testing purposes. It is worth noting that the range of different poses in the training set was limited as each video depicts just one action.

As the results show, the behavior of the classifiers is similar to that in the previous experiment. In this case, the overall accuracy increased slightly in every case despite this experiment having 10 different categories and the previous one only 4. The best performer in this case was also the SVM, which outperformed the KNN and the RF for every frameskip setting. Overall, it could be appreciated that the accuracy of RF decreased as the number of samples also decreased. This behaviour was also exhibited by the KNN, but the drop was not so considerable. The SVM performs similarly across all the experiments. DT and NB are far behind in terms of accuracy.

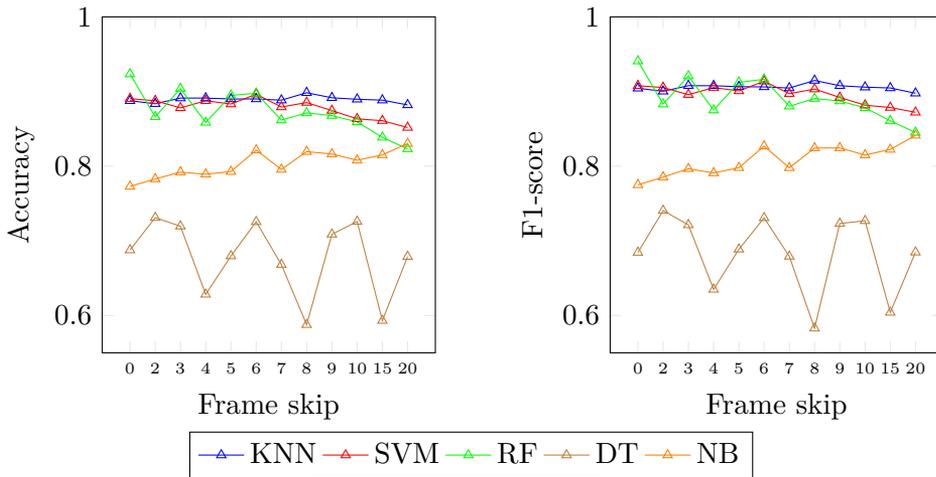


Figure 5.28: Accuracy and F1-score results for OUR's testing videos using VGG16 with frame skip from 0 to 20.

These conclusions can be extrapolated to the experiments that involved VGG16 as the backbone, as shown in Figure 5.33. In this case, the trends remained the same but with a lower overall accuracy. This is for two main reasons: on the one hand, the feature vector it provides has 25088 parameters. Despite some works concluding that the number of parameters may not impact on the accuracy of the classifiers, in this case it definitely did. On the other hand, VGG16 provided lower classification accuracy than ResNet50 when it comes to the fully convolutional network including the last fully connected layer, so the features were also likely poorer.

The experiments that used MobileNetV2 as the backbone also show the same trend. The overall accuracy was better than that of the VGG16 but poorer than the ResNet50 approach. This is because MobileNetV2 was purposely designed to be very fast to predict with, so the classification accuracy was lower than that of both the other mentioned networks. Nonetheless, as the number of features in its feature map was significantly smaller than the VGG16, its accuracy was better.

Figures 5.27, 5.28, 5.29, 5.30, 5.31 and 5.32 show the F1-score for each experiment. As can be seen, there is no bias towards a certain category.

To enable the system to perform in real environments, it should learn as fast as possible. With the goal of benchmarking this, the total time

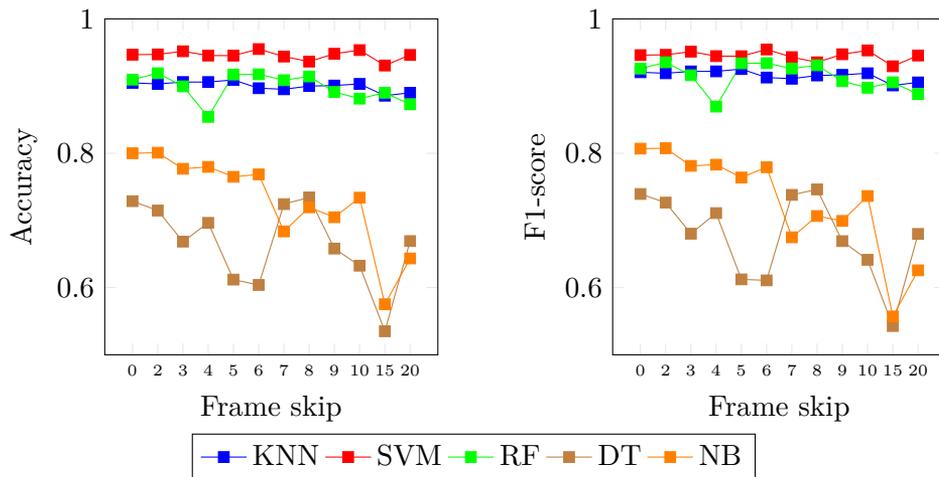


Figure 5.29: Accuracy and F1-score results for OUR's testing videos using MobileNetV2 with frame skip from 0 to 20.

consumed to generate each model with the different number of frames was calculated. The results are shown in Figure 5.35

Although SVM and RF had better precision performances, they consumed a lot of time of training compared to KNN, which is almost immediate in every case (KNN is considered lazy learning indeed). 300 seconds for training (or 64 in the case of RF) takes as much time for a real application problem as for only 1200 frames. As expected, the approaches that involved VGG16 as the backbone took a vast amount of time to train because of the number of features. Regarding ResNet50 and MobileNetV2, the training times were similar, but MobileNetV2 was slightly faster. In addition, the training time grew as the problem became more complex. For instance, all the experiments that involved the KARD dataset took longer than the experiments on the proposed one. This was also expected as the KARD dataset features 10 different classes and the proposed one only 4.

In view of these results, it can be stated that the most suitable setting for online learning purposes involves ResNet50 or MobileNetV2 as the feature extractor and KNN as the final classifier. This is the fastest and most accurate setup. As the ResNet50 is slightly more accurate, it was selected for the following experiment.

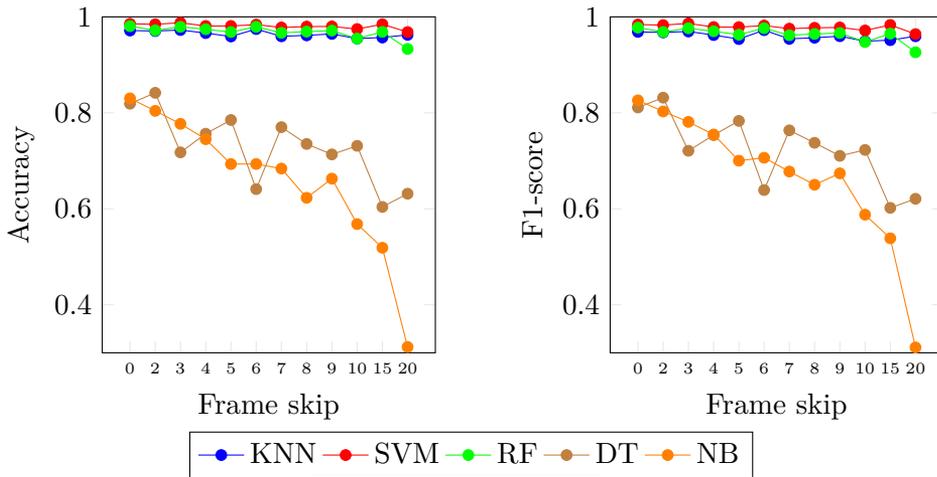


Figure 5.30: Accuracy and F1-score results for KARDs testing videos using ResNet50 with frame skip from 0 to 20.

For the second experiment, the ability of the proposed system to distinguish between known and unknown subjects was tested. To do this, a set of different classifiers (C1) to separate between these two classes was tested. If the person was classified as known, the KNN classifier (C2) from the previous experiment was used to perform the recognition. This was an important feature because the robot was likely to find new persons that have not as yet been considered by its model. The system should, thus, recognize when a person is new before trying to classify it into the identities it already knows.

In this experiment, the dataset described in Section 5.4.2.1 was used for training. The frameskip was set to 10 seconds with a frame skip of 10 (1 frame selected of every 10). The test set was made up of videos with some people from the training set and others that were not. The results are shown in Table 5.4. The ACK value represents the accuracy of the second classifier C2 after C1 has classified the person as known. The ACU value represents the accuracy of the first classifier C1 to classify unknown examples.

According to the results obtained, the best performance trade-off was achieved by the Clustering-based Local Outlier Detector because it shows high accuracy with known and unknown examples. In this case, the results

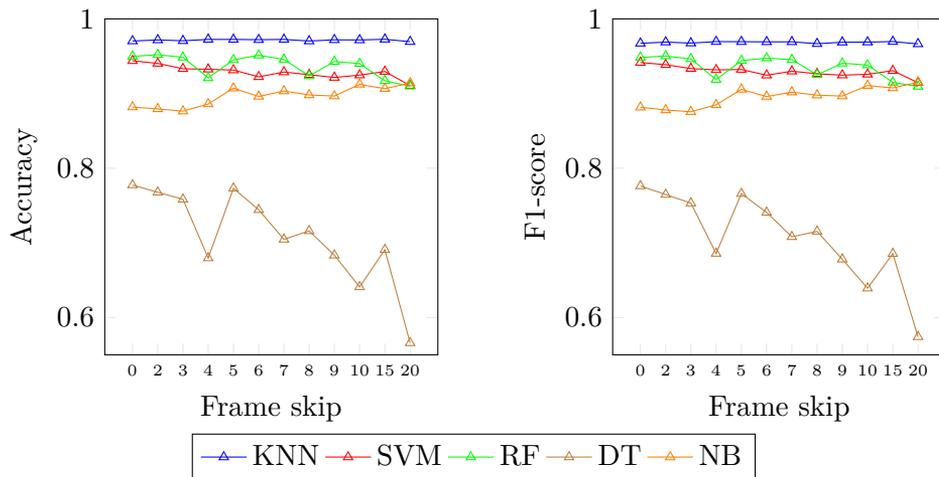


Figure 5.31: Accuracy and F1-score results for KARD's testing videos using VGG16 with frame skip from 0 to 20.

for ACU show a very high accuracy for unknown people detection without losing too much accuracy on known examples. The training time for this first classifier was 1.21 seconds, so it can be retrained almost once per second and is suitable for a fast application. Another interesting choice would be the Local Outlier Factor. This method has lower accuracy classifying unknown examples (ACU) but the ACK value shows a better precision with known subjects. However, the notable difference in ACU value (13% lower compared with the previous method versus 5% higher in ACK) shows that the Clustering-based Local Outlier Detector performance is more balanced. Nevertheless, the model time of the second method was only 0.05 seconds, so it could be used if the time requirement gets tougher. The vast majority of the other classifiers show a biased classification, with a large number of the examples classified as known or unknown only, as can be seen in the ACU accuracy results. In Figure 5.36 a random sample is shown for qualitative evaluation of the whole system. In the first example, it can be seen that the body detector has identified a reflection on the glass as a person, but the proposed system has managed to identify it as unknown. The following examples show the potential problems of the system, the partial or total occlusion of the body. These examples will be discussed later in Section 5.4.2.4.

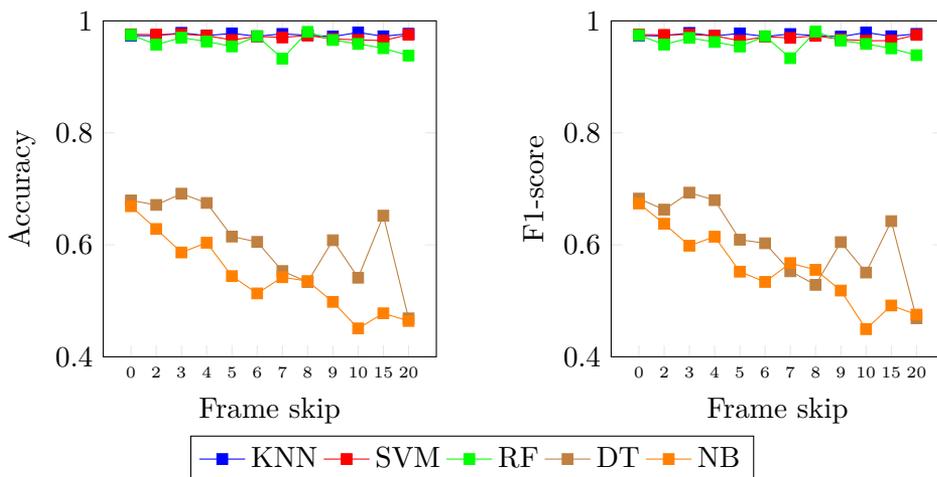


Figure 5.32: Accuracy and F1-score results for KARD's testing videos using MobileNetV2 with frame skip from 0 to 20.

Table 5.4: Results for the known-unknown classifiers and the whole recognition system

Algorithm	ACK (%)	ACU (%)	Model time (s)
Angle-based Outlier Detection [Kriegel et al., 2008]	84.86	37.87	0.95
Clustering-based Local Outlier Detector [He et al., 2003]	78.84	89.89	1.21
Histogram-based Outlier Detection [Goldstein and Dengel, 2012]	91.27	0.45	1.96
Isolation Forest [Liu et al., 2008]	90.31	11.17	0.28
Local Outlier Factor [Breunig et al., 2000]	83.30	76.36	0.05
Minimum Covariance Determinant [Rousseeuw and Driessen, 1999]	40.00	99.99	97.63
One-class SVM [Chen et al., 2001]	86.37	16.11	0.03
Principal Components Analysis [Wold et al., 1987]	82.15	23.91	0.04
Stochastic Outlier Selection [Janssens et al., 2012]	92.74	0.00	0.53

A video of this setup can be seen at ¹. First, the target subject was recorded and its features extracted (frameskip=10) and appended to both C1 (Clustering-based Local Outlier Detector) and C2 (KNN) models. The backbone was ResNet50. This setup was that previously worked best in terms of accuracy and training time. Then, the pipeline was used for tracking the target within a crowded street. The video was not edited at all, that is, between the training and the testing videos both models were trained live. It is important to notice that the results obtained with the classifiers are aimed at showing how well they perform on the same features (the same DL networks), and not to state that other classifiers could not

¹<https://youtu.be/c1biTDNnLsg>

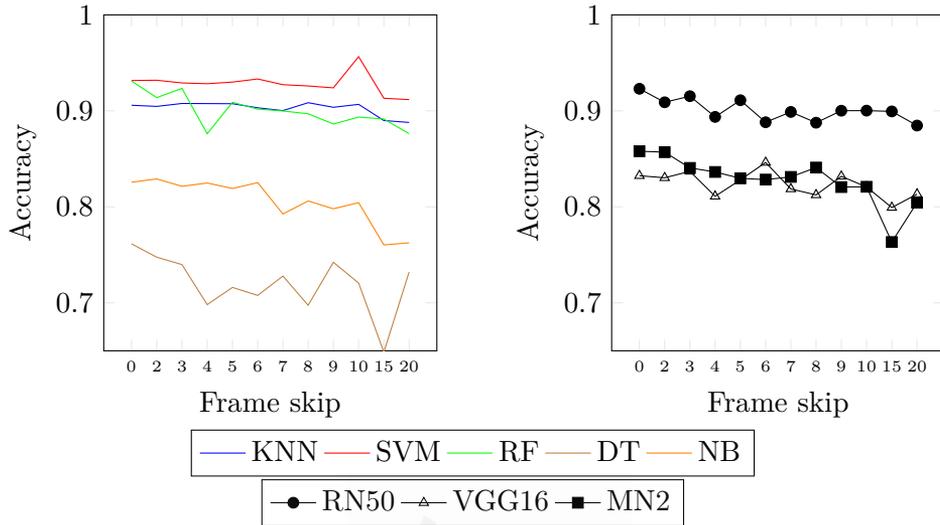


Figure 5.33: Evolution of the mean accuracy results for the classifiers and DL networks. Tested using testing videos of OUR dataset with frame skip from 0 to 20.



Figure 5.34: Random results of the proposed approach with the bounding box and the predicted identification of the person superimposed. Boxes and texts in green mean a hit (in this case, every prediction is correct). The model for these results was generated using the full duration of the training videos with no frame skip. Note that the identification is accurate even in unconsidered poses.

obtain similar results if they were applied with different configurations properly set.

5.4.2.4 Limitations

Despite the high accuracy in the test scenario, the proposed approach has some limitations. For instance, it is highly dependent on the visual features present in the training data. This means that if the person is not

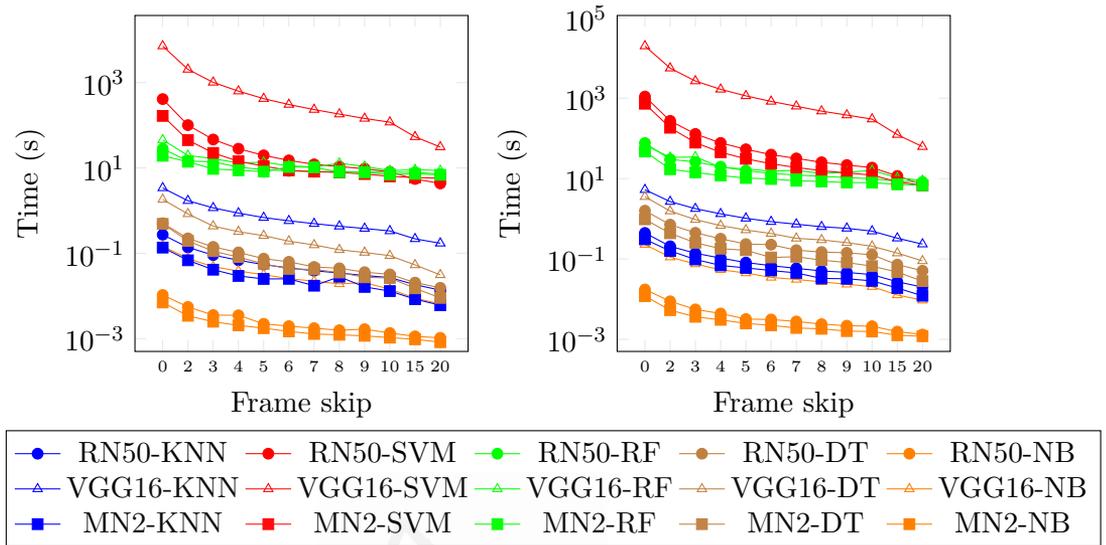


Figure 5.35: Time consumed to generate the models from OUR's and KARD's training videos using frame skip from 0 to 20.



Figure 5.36: Random results of the proposed approach with the bounding box and the predicted identification of the person superimposed. Boxes and texts in green mean a hit with known people. Blue boxes and texts mean they have been classified as unknown.

properly represented in the model, the system is likely to fail. This also makes the system fail under high occlusion scenarios. Even if the AOI of the person is correctly detected, the visual features would depict the object occluding the person, leading to an eventual error. In addition, our approach is constrained to work on one camera and with specific conditions. For instance, our proposal could not be deployed for re-identification of the same person across different surveillance cameras because they would depict different points of view, usually show a large number of persons and feature low resolution and are very noisy.

5.4.3 Conclusion and future works

This section presents PIMS, a person identification system. Such a is critical in social robots since a robot can thus learn on the fly to recognize different people and adapt its behavior to each of them. Its impact on social applications is high and can be applied to interaction in highly populated environments or care applications.

This system performs person identification using a combination of DL and traditional methods which can learn fast and live. The crop and the features of every person are extracted with DL methods and are then classified with traditional techniques.

Based on the experiments carried out, the proposed approach is able to correctly state the identification of a person in more than the 80% of the cases with only 10 seconds of training data, which perfectly suits the characteristics of the RoboCup challenge presented. Additionally, the results suggest that the accuracy of the model is independent of the number of samples. In fact, it is desirable to have a light model with different postures with a high variability rather than a lot of samples that are highly similar to one another.

Furthermore, as stated in the limitations section, the results suggest that this approach tends to fail with occluded bodies or with poses that differ from those used for training. The other possible problem is the detector not segmenting the person properly.

As a future work, it is planned to improve the PIMS accuracy by integrating a tracking method. For instance, if an identification in a certain moment differs from the last n predictions, it is likely a failure and could be corrected. In addition, face recognition must be included to complement and enhance the performance of the system. We also want to try newer classification algorithms, like [Rafiei and Adeli, 2017, Ahmadlou and Adeli, 2010].

Conclusions

The conclusions of this work are given in this chapter. Section 6.1 contains the general conclusions on the work carried out in the thesis. The main contributions of this thesis are listed in Section 6.2. The publications that were produced during the preparation of the thesis are collected in Section 6.3. Finally, Section 6.4 explains future lines of research to improve and extend the work carried out in this thesis.

6.1 Conclusions

In this thesis, different alternative methods for the realisation of 3D object recognition have been proposed.

In the case of NurbsNet, similarity between free forms, nurbs in this case, has been introduced for the first time to determine the global classification of a cloud of points representing an object. To do this, research had to be done on a distance metric that would allow two surfaces to be properly compared in terms of shape similarity. Despite not surpassing the state of the art, this work opens up new, unpublished avenues of research for more organic object recognition that can be more easily understood by humans.

In the case of VFD, the mathematical concept of the fractal dimension has been used for the first time for the recognition of three-dimensional objects. The results of this method are promising and invite further research on fractals applied to object recognition and other fields of 3D data

processing.

It is worth noting that the methods used make use of the point cloud as input, which is a plus since this is the representation chosen by the vast majority of three-dimensional sensors.

Regarding the mapping of the environment, we have proposed the development of an automatic 3D environment mapping system in which not only the points are included directly in their corresponding location, but also additional processing has been carried out to give meaning to these points. In our research we have provided these maps of the environment with the recognition and segmentation of the objects present in the scene, as well as the detection of potentially dangerous areas for people.

Likewise, we have proposed a planning system that allows to calculate routes between rooms using a semantic map of the environment, and returns the most optimal route between rooms taking into account connectivity criteria (corridors and open doors) and cost (distance, time or other circumstances). This system allows the navigation of a social robot between rooms.

In addition, we have proposed a method for merging three-dimensional data from various sources, which allows us to take advantage of the benefits of each source and overcome their disadvantages.

Regarding robotics, we have conducted a study on the state of social robotics concerning the care of the elderly, active aging and therapies with autistic people. In this way, we have been able to know where we are and propose solutions to existing problems.

Based on this previous research, we have proposed an augmented reality system that allows to help in physical rehabilitation tasks within a home, and to be evaluated quantitatively with our metrics, so that the therapist can know the evolution of the therapy.

Finally, we have presented a human recognition module capable of learning new identities in real time, which would allow robots to treat their interlocutors in a personalized way.

6.2 Contributions of the thesis

The contributions made in this thesis are as follows:

- *NurbsNet* and *Voxelized Fractal Descriptor* are two architectures for 3D object recognition from point clouds using free-form surfaces and fractal dimension respectively.
- Automatic 3D mapping of the environment including information about objects in the environment and the delimitation of hazardous areas.
- A route planning system based on the connectivity between rooms within a semantic map, taking into account the cost (time or other circumstances) of travelling between rooms.
- A method for combining information from a depth sensor with depth predictions from monocular images, taking advantage of the benefits of each and overcoming their disadvantages. In this way, the problems inherent in the Pepper robot's depth sensor have been corrected.
- A review of the current state of social robotics focused on the field of care for the elderly and people with autism, which gives an idea of the current state of robot-assisted rehabilitation therapies.
- A method for the evaluation of poses and rehabilitation exercises for physical therapy within the patient's home environment.
- An appearance-based person re-identification module, integrable within a social robot, capable of learning new identities and performing real-time inference.

6.3 Publications

During this thesis, 7 works have been published in high impact journals, rated by the Journal Citation Reports (Journal Citation Reports (JCR)). In addition, 3 contributions have been presented at international conferences.

Specifically, the following articles were published as a result of the research carried out during this thesis.

Published articles in scientific journals:

- Cruz, E., Escalona, F., Bauer, Z., Cazorla, M., García-Rodríguez, J., Martínez-Martin, E., and Gomez-Donoso, F. (2018). Geoffrey: an automated schedule system on a social robot for the intellectually challenged. *Computational intelligence and neuroscience*, 2018. [Cruz et al., 2018a] (**Chapter 4.4**)
- Gomez-Donoso, F., Escalona, F., Rivas, F. M., Cañas, J. M., and Cazorla, M. (2019). Enhancing the ambient assisted living capabilities with a mobile robot. *Computational intelligence and neuroscience*, 2019. [Gomez-Donoso et al., 2019] (**Chapter 4.3**)
- Escalona, F., Martínez-Martin, E., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). EVA: EVALuating at-home rehabilitation exercises using augmented reality and low-cost sensors. *Virtual Reality*, 1-15. [Escalona et al., 2019] (**Chapter 5.3**)
- Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). Refining the Fusion of Pepper Robot and Estimated Depth Maps Method for Improved 3D Perception. *IEEE Access*, 7, 185076-185085. [Bauer et al., 2019] (**Chapter 4.5**)
- Martínez-Martin, E., Escalona, F., and Cazorla, M. (2020). Socially assistive robots for older adults and people with autism: An overview. *Electronics*, 9(2), 367. [Martínez-Martin et al., 2020] (**Chapter 5.2**)
- Martín-Rico, F., Gomez-Donoso, F., Escalona, F., Garcia-Rodriguez, J., and Cazorla, M. (2020). Semantic visual recognition in a cognitive architecture for social robots. *Integrated Computer-Aided Engineering*, (Preprint), 1-16. [Martin-Rico et al., 2020] (**Chapter 5.4**)
- Domenech, J. F., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2020). A Voxelized Fractal Descriptor for 3D Object Recognition. *IEEE Access*, 8, 161958-161968. [Domenech et al., 2020] (**Chapter 3.3**)

International conferences:

- Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2017, November). 3D Object Mapping Using a Labelling System. In Iberian Robotics conference (pp. 579-590). Springer, Cham. [Escalona et al., 2017a] **(Chapter 4.2)**
- Martin-Rico, F., Gomez-Donoso, F., Escalona, F., Cazorla, M., and Garcia-Rodriguez, J. (2019, June). Artificial Semantic Memory with Autonomous Learning Applied to Social Robots. In International Work-Conference on the Interplay Between Natural and Artificial Computation (pp. 401-411). Springer, Cham. [Martin-Rico et al., 2019] **(Chapter 5.4)**
- Escalona, F., Viejo, D., Fisher, R. B., and Cazorla, M. (2020, July). NurbsNet: A Nurbs approach for 3d object recognition. In 2020 International Joint Conference on Neural Networks (IJCNN) (pp. 1-7). IEEE. [Escalona et al., 2020] **Selected as a finalist (5 candidates) for the IJCNN-IEEE 2020 Best Student Paper Award (Chapter 3.2)**

Other works to which contributions have been made during the course of the thesis but whose content has not been included in the thesis:

- Escalona, F., Rodriguez, A., Gomez-Donoso, F., Martinez-Gomez, J., and Cazorla, M. (2017). 3D object detection with deep learning. *Journal of Physical Agents*, 8(1), 3-10. [Escalona et al., 2017c]
- Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2018, November). Improving the 3D perception of the pepper robot using depth prediction from monocular frames. In *Workshop of Physical Agents* (pp. 132-146). Springer, Cham. [Bauer et al., 2018]
- Garcia-Rodriguez, J., Gomez-Donoso, F., Oprea, S., Garcia-Garcia, A., Cazorla, M., Orts-Escolano, S., ... and Rivas-Montero, F. (2020). COMBAHO: A deep learning system for integrating brain injury

patients in society. *Pattern Recognition Letters*, 137, 80-90. [Garcia-Rodriguez et al., 2020]

- Escalona, F., Gomez-Donoso, F., Viejo, D., Orts-Escolano, S., and Cazorla, M. (2017). PRACTICAL CLASSES IN COMPUTER PROGRAMMING FOR A ROBOTICS ENGINEER. In *INTED2017 Proceedings* (pp. 2486-2490). IATED. [Escalona et al., 2017b]
- Torres-Camara, J. M., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2019, November). Map Slammer: Densifying Scattered KSLAM 3D Maps with Estimated Depth. In *Iberian Robotics conference* (pp. 563-574). Springer, Cham. [Torres-Camara et al., 2019]
- Gomez-Donoso, F., Escalona, F., and Cazorla, M. (2020). Par3dnet: Using 3dcnns for object recognition on tridimensional partial views. *Applied Sciences*, 10(10), 3409. [Gomez-Donoso et al., 2020b]
- Gomez-Donoso, F., Escalona, F., Nasri, N., and Cazorla, M. (2021). A Hand Motor Skills Rehabilitation for the Injured Implemented on a Social Robot. *Applied Sciences*, 11(7), 2943. [Gomez-Donoso et al., 2021a]
- Gomez-Donoso, F., Escalona, F., Bañuls, A., Abellan, D., and Cazorla, M. (2020, November). Monocular 3D Hand Pose Estimation for Teleoperating Low-Cost Actuators. In *Workshop of Physical Agents* (pp. 345-359). Springer, Cham. [Gomez-Donoso et al., 2020a]
- Gomez-Donoso, F., Escalona, F., Pérez-Esteve, F., and Cazorla, M. (2021). Accurate Multilevel Classification for Wildlife Images. *Computational Intelligence and Neuroscience*, 2021. [Gomez-Donoso et al., 2021b]

6.4 Future work

As for 3D object recognition, improvements are planned for the two approaches we have proposed.

For recognition using free-form shapes, we are investigating how best to incorporate the updating and optimisation of the surfaces embedded in the Nurbs layer to automatically obtain those surfaces that best differentiate objects, analogous to how convolution filters learn their weights to highlight the features that best represent the dataset, which may allow us to both improve classification results and better understand the network's decision-making process.

As for object recognition using fractals, we plan to create a local descriptor based on the same fundamentals as this descriptor, so that we are able to uniquely describe each point in a point cloud, which would allow us to do object recognition using incomplete models.

Regarding the mapping of the environment, we plan to model the obstacles taking into account that the environment can be changeable, trying to separate the fixed structures of each room (walls, floor) from other elements that can be subject to change, such as chairs, tables or shelves. In this way, the robot would detect changes in the environment and appropriately update the information on both the structure of the scene and the objects present in the environment.

As for the person re-identification system, our aim is to incorporate facial recognition following a similar strategy to that followed in the implemented body recognition, so that the robot can still recognise people conveniently even if they make changes in their clothing.



Universitat d'Alacant
Universidad de Alicante

The logo of the University of Alicante, featuring a stylized 'A' with horizontal bars inside, set within a square frame.

Appendices

Universitat d'Alacant
Universidad de Alicante

Introducción

Este capítulo presenta los temas de esta tesis. El capítulo está organizado como sigue. En la sección A.1 se presenta la motivación de este trabajo. A continuación, en la sección A.2, se muestran las diferentes herramientas utilizadas durante el desarrollo de la tesis. En la Sección A.3, se presenta la propuesta y los objetivos. Finalmente, en el apartado A.4 se muestra la estructura de la tesis.

A.1 Introducción y motivación

La presente tesis aborda tres temas principales. En primer lugar, se trata el reconocimiento de objetos en 3D sobre nubes de puntos. A continuación, se trata el mapeo del entorno. Por último, se incluyen otros temas relacionados con la robótica social.

Los métodos de reconocimiento de objetos tridimensionales se centran en la clasificación de objetos representados en 3 coordenadas, al igual que en el mundo real. Estos métodos se diferencian de sus homólogos en 2D en que se centran en los aspectos topológicos de los datos y no en su aspecto visual. Aunque el reconocimiento en 2D tiene una alta tasa de aciertos y es perfectamente aplicable en muchas situaciones, hay situaciones en las que no funciona correctamente. Por ejemplo, tiene grandes dificultades para diferenciar objetos que tienen un aspecto muy similar, como diferenciar una fotografía de un objeto real. Por ello, el reconocimiento de objetos en 3D puede complementar y superar estas circunstancias. Por un lado, los

métodos tradicionales que realizan esta tarea son muy dependientes de las condiciones del conjunto de datos, ya que se ven muy perjudicados por las oclusiones y el ruido, y a veces carecen de generalizabilidad. Por otro lado, existen varios métodos que se basan en el aprendizaje profundo para realizar la clasificación. A pesar de sus buenos resultados, siguen sufriendo en el ámbito de la explicabilidad, es decir, la facilidad con la que sus decisiones pueden ser entendidas por un ser humano. Exploraremos nuevos métodos para realizar esta clasificación que puedan ser más orgánicos y comprensibles.

En cuanto al mapeado del entorno, estos sistemas abordan el problema de la adquisición de modelos espaciales de entornos físicos que puedan ser utilizados por robots móviles. Como el objetivo de este trabajo es que lo utilice un robot social, nos centraremos únicamente en las técnicas de mapeo de interiores. Existen métodos buenos y robustos que asumen que el entorno es estático, estructurado y de tamaño limitado. Sin embargo, trabajar con mapas del entorno a gran escala, no estructurados y dinámicos, es un problema que aún está lejos de ser resuelto. Históricamente, la investigación se ha dividido entre mapas métricos y topológicos. Los mapas métricos almacenan las propiedades geométricas del entorno, mientras que los mapas topológicos describen las conexiones entre los distintos lugares del entorno. En esta tesis, trataremos de combinar las ventajas de estos dos tipos de mapas mapeando el entorno para incluir información semántica sobre los objetos de la escena. De este modo, un robot podrá navegar por el entorno e interactuar adecuadamente con aquellos objetos que necesite para realizar su tarea programada.

En cuanto a la robótica social, en esta tesis incluimos una revisión exhaustiva del estado del arte de los robots sociales, especialmente centrada en la interacción con personas mayores y personas con autismo. Según la Organización Mundial de la Salud, las personas con discapacidades son especialmente vulnerables a las deficiencias de los servicios, como la atención sanitaria, la rehabilitación, el apoyo y la asistencia. En este sentido, los recientes desarrollos tecnológicos pueden mitigar estas deficiencias, ofreciendo sistemas de asistencia menos costosos para satisfacer las necesidades de los usuarios. Por otro lado, en este trabajo proponemos un sistema de

realidad aumentada para ayudar en la adherencia y supervisión de las sesiones de rehabilitación en casa, que puede realizarse con sensores de bajo coste y, por tanto, ser accesible a un gran número de pacientes. Este sistema almacena las estadísticas del usuario y permite a los terapeutas ajustar los ejercicios en función de los resultados obtenidos. Por último, en esta tesis presentamos un módulo de reconocimiento visual que permite a los robots identificar a las personas con las que interactúan para ofrecerles un tratamiento personalizado. La idea principal de este trabajo es que el robot sea capaz de aprender nuevas identidades en tiempo real con una pequeña interacción con el usuario, y a partir de ese momento poder reconocerlo inequívocamente en otras ocasiones.

Es importante destacar que esta tesis se ha realizado en el marco de los siguientes proyectos:

- *RETOGAR: Un sistema para potenciar la autonomía de los lesionados cerebrales y discapacitados para su integración en la sociedad.* Financiado por el Ministerio de Economía de España y apoyado por fondos FEDER. DPI2016-76515-R [Garcia-Rodriguez et al., 2020].
- *A2HUMPA: Aprendizaje y análisis de comportamientos humanos para monitorización, asistencia personalizada y detección temprana de dolencias.* Financiado por el Ministerio de Ciencia e Innovación de España y apoyado por fondos FEDER PID2019-104818RB-I00.

Además, con las siguientes ayudas:

- *Beca FPU para estudios de doctorado.* Con el apoyo del Gobierno de España, Ministerio de Ciencia e Innovación. FPU16/00887.

A.2 Metodología

Este capítulo incluye una explicación de las herramientas, tanto de hardware como de software, utilizadas en el proyecto, así como los frameworks y conjuntos de datos utilizados para entrenar y validar los resultados.

A.2.1 ASUS Xtion Pro Live



Figura A.1: Sensor ASUS Xtion Pro Live. Obtenido de [McGlaun, 2011]

Se trata de una cámara 3D dirigida principalmente a los desarrolladores. Consta de un sensor de infrarrojos, micrófonos para captar el sonido ambiente y una cámara RGB que permite colorear la imagen de profundidad.

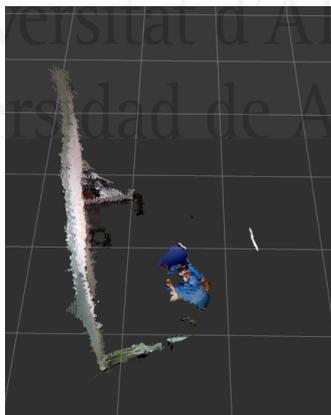


Figura A.2: Vista de una nube de puntos obtenida de Xtion registrada con la información de color.

Su resolución máxima es de 640x480, el rango de profundidad que detecta está entre 0,8 y 3,5 metros y su campo de visión es de 58° en horizontal y 45° en vertical.

A.2.2 Robot Pepper

Pepper es un robot humanoide creado por Aldebaran Robotics y Soft-Bank diseñado para ser un robot social que analiza expresiones y tonos de voz. Se presentó en una conferencia en junio de 2014 y se comercializó en febrero de 2015. Como se ve en la Figura A.4, el robot tiene una altura de 1,21 metros, una anchura de 480 mm y una profundidad de 425 mm.

Su cabeza tiene 4 micrófonos (Figura A.5(a)), dos altavoces (Figura A.5(b)), dos cámaras RGB -en la boca y en la frente (Figura A.5(c)- y un sensor de profundidad Xtion detrás de los ojos (Figura A.5(d)). Tiene un giroscopio en el torso y sensores táctiles en la cabeza y el dorso de las manos. Su base móvil consta de 2 sonares (Figura A.5(e)), 6 láseres (Figura A.5(f)), 3 sensores de choque y un giroscopio. Gracias a estos elementos, puede percibir adecuadamente el entorno que le rodea para interpretarlo y actuar en consecuencia.

En cuanto a la parte motriz del robot, al tratarse de un robot humanoide, dispone de las mismas articulaciones que una persona, salvo la imposibilidad de mover los dedos de forma independiente y la sustitución de las extremidades inferiores por una base móvil, tal y como se muestra en la Figura A.6. De esta forma, la estabilidad respecto a otros robots humanoides bípedos está garantizada, facilitando su movilidad.

Dada una articulación que conecta dos partes del robot, se considera que la parte del robot más cercana al tronco está fija y que la parte más alejada gira alrededor del eje de la articulación. Para cada una de las articulaciones se establece un sistema de coordenadas de forma que en la posición 0 del robot todas tengan la misma orientación. La Figura A.7 muestra la rotación correspondiente a *roll*, *pitch* y *yaw* según la convención establecida.

Las siguientes Figuras muestran el rango de rotación de la cabeza (Figura A.8), las articulaciones de la extremidad izquierda (Figura A.9) y derecha (Figura A.10) de las articulaciones de las extremidades y de la cadera (Figura A.11).

En las manos, el robot dispone de 5 dedos (Figura A.12) pero sólo puede abrirlos o cerrarlos debido a la presencia de un único motor por mano.



Figura A.3: Robot Pepper del laboratorio de investigación de Rovit

Universitat d'Alacant
Universidad de Alicante

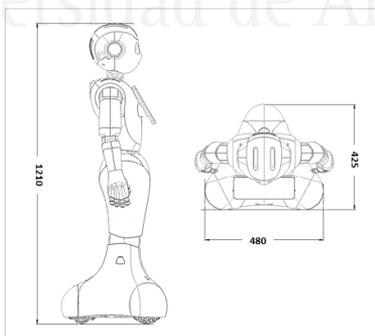


Figura A.4: Dimensiones del Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

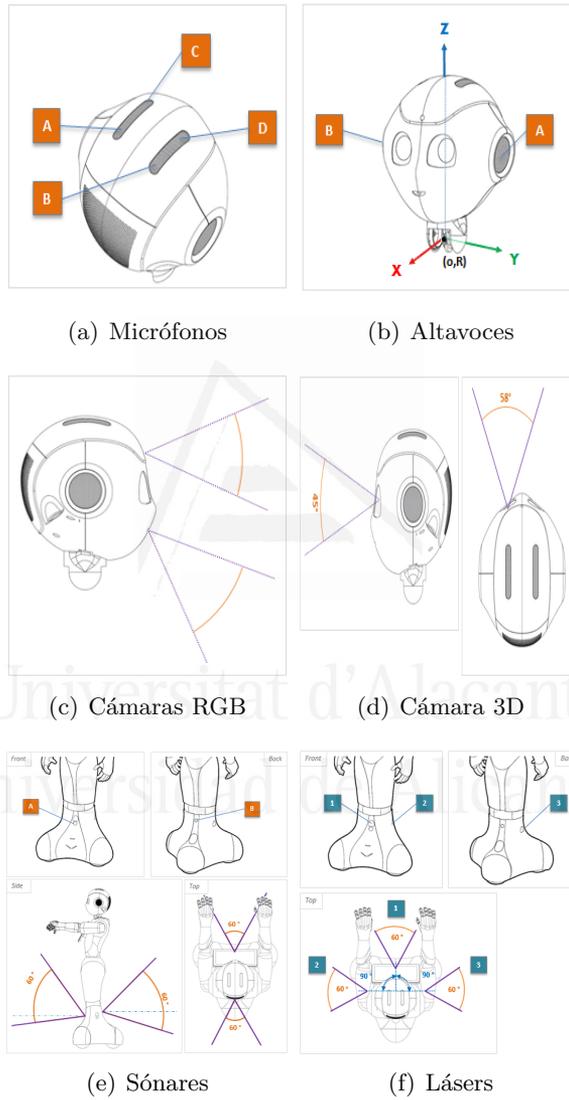


Figura A.5: Sensores y actuadores de Pepper. Obtenido de [Aldebaran-Robotics, 2014]

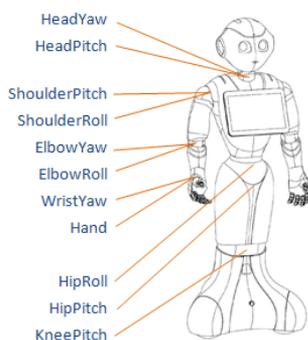


Figura A.6: Articulaciones del Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

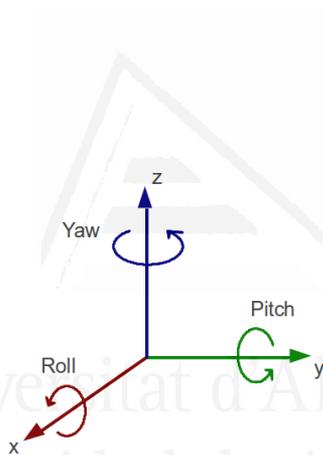


Figura A.7: Convención de signos para los sistemas de coordenadas del Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

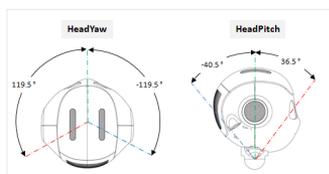


Figura A.8: Rango de giro de la cabeza del Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

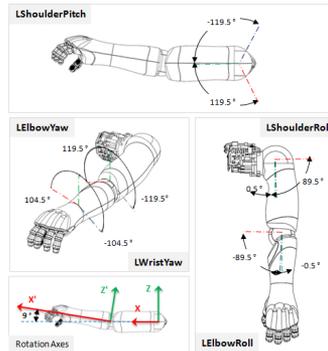


Figura A.9: Rango de giro de la extremidad izquierda del Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

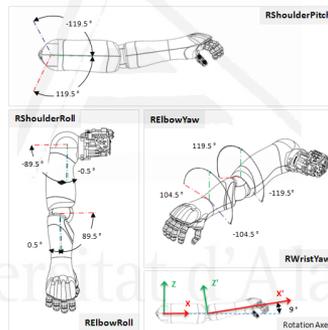


Figura A.10: Rango de rotación de la extremidad derecha del robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

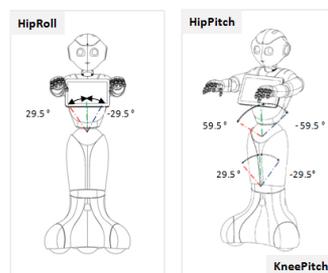


Figura A.11: Rango de oscilación de la cadera de Robot Pepper. Obtenido de [Aldebaran-Robotics, 2014]

los se comunican entre sí enviando mensajes a servicios ofrecidos por otros módulos y recibiendo su respuesta.

Una tercera posibilidad para programar el robot Pepper es trabajar con la interfaz ROS desarrollada por su comunidad. Esta interfaz proporciona un puente entre *NAOqi* y ROS, transformando los mensajes del primero en mensajes que pueden ser utilizados por el segundo [ROS, 2014].

A.2.2.1 El problema del sensor 3D del robot Pepper

La mayoría de los experimentos se realizaron con el robot Pepper y muchos métodos utilizan la información 3D de la cámara de profundidad. El robot tiene tres versiones de hardware diferentes. La última versión 1.8 incluye un nuevo sensor tridimensional. En la versión más reciente del robot, el sensor de profundidad fue sustituido por dos cámaras de color de 4 MPx, y se utilizó un algoritmo estéreo para proporcionar la percepción de la profundidad. Los mapas de profundidad resultantes se generan a 15 fps y tienen una resolución de 1280×720 . Sin embargo, las versiones 1.6 y 1.8a, están equipadas con el sensor de profundidad defectuoso, Asus Xtion. El sensor de profundidad Asus Xtion puede proporcionar mapas de profundidad con una resolución de 320×240 a 20 cuadros por segundo.

Como se ha mencionado, el Pepper Robot tiene un sensor Asus Xtion como cámara de profundidad. En teoría, esta cámara puede proporcionar mapas de profundidad precisos de las escenas. Sin embargo, la cámara Xtion montada en este Robot parece proporcionar mapas de profundidad erróneos, lo que también da lugar a nubes de puntos incorrectas.

Como se muestra en la Figura A.14, este problema radica en una representación distorsionada del espacio tridimensional. Las nubes de puntos resultantes revelan un patrón ondulado en toda la escena. Este problema se hace más evidente cuando se representan artefactos planos, como paredes o suelos, pero se produce en toda la escena. Además, observamos que la distorsión empeora al aumentar la profundidad, es decir, que los objetos cercanos al sensor se ven menos afectados que los más alejados.

Hay que señalar que no se trata de un problema aislado ni de un defecto de nuestro dispositivo. Para asegurarnos de ello, hemos contactado personalmente con diferentes investigadores de distintos laboratorios y univer-

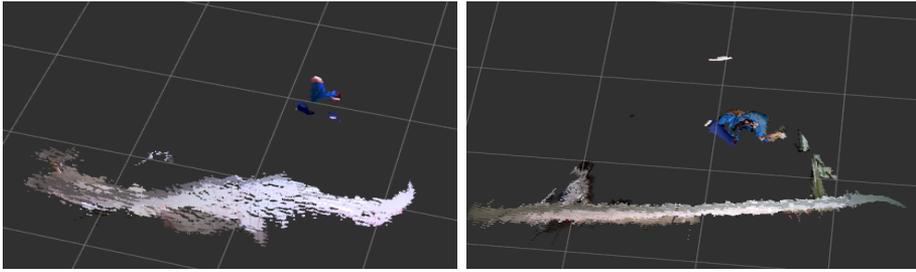


Figura A.14: Estas imágenes muestran una nube de puntos Xtion externa e independiente y la cámara Xtion del Robot Pepper. Ambas imágenes están tomadas desde el mismo punto de vista y representan la misma escena. El artefacto blanco es una superficie plana que, en este caso, es una pared.

sidades que han informado del mismo problema con el robot con la misma versión.

A.2.3 ROS

ROS es un framework para el desarrollo de software para robots que funciona como un sistema operativo dentro de un cluster heterogéneo.

ROS proporciona los servicios de un sistema operativo como la abstracción de hardware, el control de dispositivos de bajo nivel o el paso de mensajes entre procesos. Está diseñado bajo una arquitectura de grafos en la que el procesamiento está distribuido y se realiza en cada uno de los nodos, que pueden recibir y enviar diferentes tipos de mensajes según sus necesidades.

ROS ha sido desarrollado bajo una licencia de software libre BSD, que permite su uso comercial y de investigación de forma gratuita, y cuenta con una amplia comunidad de desarrolladores que aportan paquetes de software con diferentes utilidades para la robótica. [Quigley et al., 2009b]

A.2.4 Gazebo

Gazebo es un simulador de dinámica 3D diseñado para reproducir con precisión los entornos dinámicos que puede encontrar un robot. Todos los objetos simulados tienen masa, velocidad, fricción y muchos otros atributos que les permiten comportarse de forma realista cuando son empujados,

tirados, derribados o transportados. Los robots son estructuras dinámicas compuestas por cuerpos rígidos conectados por articulaciones. Pueden aplicarse fuerzas, tanto angulares como lineales, a las superficies y articulaciones para generar movimiento e interacción con el entorno [Koenig and Howard, 2004].

Para lograr la integración de ROS con Gazebo, un conjunto de paquetes ROS llamados `gazebo_ros_pkgs` proporciona wrappers para conectarse con Gazebo. Proporcionan las interfaces necesarias para simular un robot en Gazebo utilizando mensajes y servicios ROS.

La Figura A.15 muestra un ejemplo de un entorno interior simulado en Gazebo.

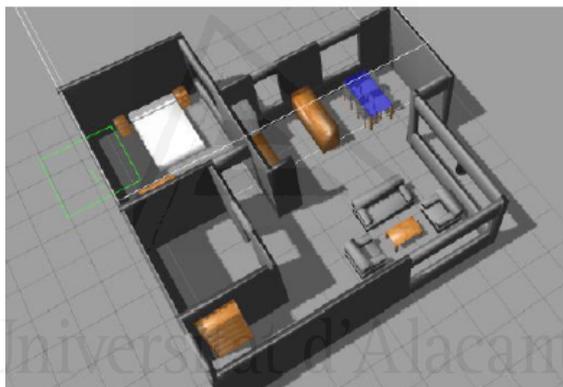


Figura A.15: Ejemplo de simulación de Gazebo. Casa de Annie. Obtenido de [Cazorla et al., 2016]

A.2.5 Gmapping

Gmapping es un algoritmo SLAM basado en láser que forma parte de OpenSLAM [Grisetti et al., 2007]. Este enfoque utiliza un filtro de partículas en el que cada partícula lleva un mapa individual del entorno. Los autores presentan técnicas adaptativas para reducir el número de partículas en un filtro de partículas Rao-Blackwellized para el aprendizaje de mapas de rejilla, teniendo en cuenta no sólo el movimiento del robot sino también la observación más reciente. La Figura A.16 muestra un ejemplo de su ejecución.

Este algoritmo se introduce en ROS en forma de wrapper en un nodo

llamado `slam_gmapping`. Utilizando este algoritmo, se pueden crear mapas de rejilla de ocupación 2-D a partir de datos láser y de pose recogidos por un robot móvil.



Figura A.16: Ejemplo de Gmapping del Campus de Friburgo. Obtenido de [Grissetti et al., 2007]

A.2.6 Caffe

Caffe es un framework de software libre, bajo licencia BSD, para el Aprendizaje Profundo, inicialmente desarrollado por el BAIR y actualmente apoyado por una gran comunidad de desarrolladores. Está desarrollado para C++, Python y Matlab.

Tiene una arquitectura expresiva que permite la innovación y la aplicabilidad. La configuración de sus modelos no requiere una programación compleja y permite el uso de GPU para acelerar el proceso de entrenamiento y reconocimiento.

El desarrollo de su código se ha realizado de forma modular, lo que ha facilitado enormemente las aportaciones de la comunidad de desarrolladores que participan en el proyecto, así como la facilidad de uso para los usuarios finales.

Caffe también proporciona modelos predefinidos y optimizados basados en conjuntos de datasets proporcionados por grandes retos de reconocimiento de objetos como ImageNet, cuyas configuraciones han tenido éxito en estos retos, resultando en altas tasas de acierto.

A.2.7 Tensorflow y Keras

TensorFlow es una plataforma integral de código abierto para el aprendizaje automático en Python y C++. Cuenta con un ecosistema completo y flexible de herramientas, bibliotecas y recursos de la comunidad que permite a los investigadores impulsar el estado del arte en ML y a los desarrolladores construir y desplegar fácilmente aplicaciones ML potenciadas [ten,]. Es uno de los frameworks DL más populares, ya que proporciona una gran facilidad de uso y permite la implementación de arquitecturas con diferentes grados de abstracción, dependiendo del nivel de experiencia del usuario.

Keras es una API diseñada para seres humanos, con un alto nivel de abstracción. Keras sigue las mejores prácticas para reducir la carga cognitiva: ofrece APIs consistentes y sencillas, minimiza el número de acciones del usuario requeridas para los casos de uso comunes, y cuenta con una amplia documentación y guías para desarrolladores [Chollet et al., 2015]. Actualmente, este sistema está totalmente integrado en TensorFlow.

A.2.8 Darknet

Darknet es un framework de redes neuronales de código abierto escrito en C y CUDA. Es rápido, fácil de instalar y admite el procesamiento de glscpu y glsgpu. Su uso no es tan popular como el de frameworks como Keras o Tensorflow, pero sigue siendo utilizado ya que algunas famosas arquitecturas de deep learning fueron implementadas con él. Es el caso de YOLO [Bochkovskiy et al., 2020], que es uno de los métodos de última generación más extendidos para el reconocimiento de objetos en imágenes.

A.2.9 PCL

PCL es una librería gratuita con licencia BSD utilizada para el procesamiento de nubes de puntos 3D. Se trata de una biblioteca escrita íntegramente en plantillas C++ con el objetivo de conseguir la mayor eficiencia y rendimiento computacional, con la mayoría de las operaciones matemáticas implementadas utilizando la biblioteca Eigen [Guennebaud et al., 2010],

una biblioteca de código abierto para álgebra lineal, y con compatibilidad con *OpenMP*.

Desde el punto de vista algorítmico, esta librería incorpora un gran número de algoritmos de procesamiento 3D que trabajan con nubes de puntos, incluyendo: filtrado, estimación de características, reconstrucción de superficies, segmentación o registro, entre otros. Cada conjunto de algoritmos está definido por clases base que intentan integrar toda la funcionalidad común, permitiendo la extensibilidad y el uso compacto y limpio de las implementaciones de los algoritmos.

PCL tiene total compatibilidad con ROS, por lo que los algoritmos de esta librería pueden desplegarse como nodelets ROS y operar como si fueran un nodo más del framework.

En cuanto a la visualización de los datos, PCL viene con su propia biblioteca de visualización, basada en VTK [Schroeder et al., 2005], que proporciona soporte para la representación de nubes de puntos 3D y formas 3D básicas [Rusu and Cousins, 2011].

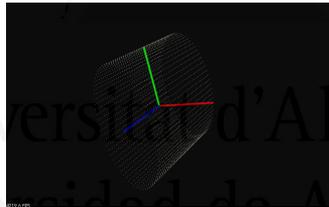


Figura A.17: Ejemplo de visualización PCL. Obtenido de [PointClouds, 2015]

A.2.10 Conjuntos de datos para reconocimiento 3D

Existen algunos conjuntos de datos del estado del arte que han capturado objetos individuales de forma aislada con sensores reales para el reconocimiento de objetos 3d, como se presenta en [Firman, 2016]. *RGBD Object Dataset* [Lai et al., 2011] y *MV-RED* [Liu et al., 2015] ofrecen capturas de 300 y 500 objetos usando Kinect v1, pero sin información de la pose. *BigBIRD dataset* [Singh et al., 2014] y *YCB Object and Model Set* [Calli et al., 2015] incorporan adicionalmente información de pose de 100 objetos utilizando un sensor Asus Xtion Pro. Sin embargo, el mayor esfuerzo ha

sido realizado por *Un gran conjunto de datos de escaneos de objetos* [Choi et al., 2016] que proporciona capturas de más de 10000 objetos utilizando un sensor PrimeSense, con la pose de la cámara calculada directamente a partir de los datos. También existen otros conjuntos de datos ampliamente utilizados, como SUNRGBD [Song et al., 2015], SCANNET [Dai et al., 2017] y S3DIS [Armeni et al., 2017]. Sin embargo, estos conjuntos de datos están destinados a la detección y segmentación de objetos, por lo que no está claro cómo adaptarlos para permitir un protocolo de prueba justo para la evaluación comparativa de los enfoques de clasificación de objetos.

El principal problema de los datos reales es la dificultad para adquirir el ground truth para la estimación de la pose en 3D, que no puede obtenerse sin hardware externo. Para resolver este problema, muchos investigadores han adoptado conjuntos de datos sintéticos. Estos conjuntos de datos permiten a los usuarios controlar más cuidadosamente varios aspectos, como la densidad de puntos, la pose o el nivel de ruido. ModelNet [Wu et al., 2015a] proporciona dos conjuntos de datos diferentes *ModelNet10* y *ModelNet40* con miles de modelos CAD para 10 y 40 clases diferentes respectivamente. Se consideran conjuntos de datos estándar de facto para tareas de reconocimiento de objetos 3D. Por otro lado, *ObjectNet* [Xiang et al., 2016], con 100 categorías y más de 40 mil formas 3D alineadas, representa otro de los mayores conjuntos de datos para este fin.

A.2.10.1 Princeton ModelNet

Princeton ModelNet [Wu et al., 2015b], es uno de los benchmarks más utilizados para el reconocimiento de objetos 3D. Este conjunto de datos tiene dos versiones: ModelNet10 y ModelNet40.

ModelNet10 ofrece un conjunto de más de 4.700 modelos CAD de 10 categorías diferentes que están alineados manualmente, y divididos en conjunto de entrenamiento y de prueba. Siguiendo los pasos explicados en [Garcia-Garcia et al., 2016], convertimos estos modelos en nubes PCD, compatibles con la librería PCL. En [Garcia-Garcia et al., 2016] también podemos ver la distribución altamente desequilibrada de los conjuntos de entrenamiento y prueba.

El principal problema de este conjunto de datos es la similitud visual

entre categorías, que se da principalmente con las categorías *mesita de noche* y *cómoda*. En la Figura A.18 podemos ver muestras de ambas categorías, que son muy difíciles, incluso para un humano, de distinguir. Está claro que hay otros diseños que son más diferenciables entre sí, pero este problema se da con bastante frecuencia, lo que conlleva una potencial disminución de la precisión de los resultados de la clasificación.

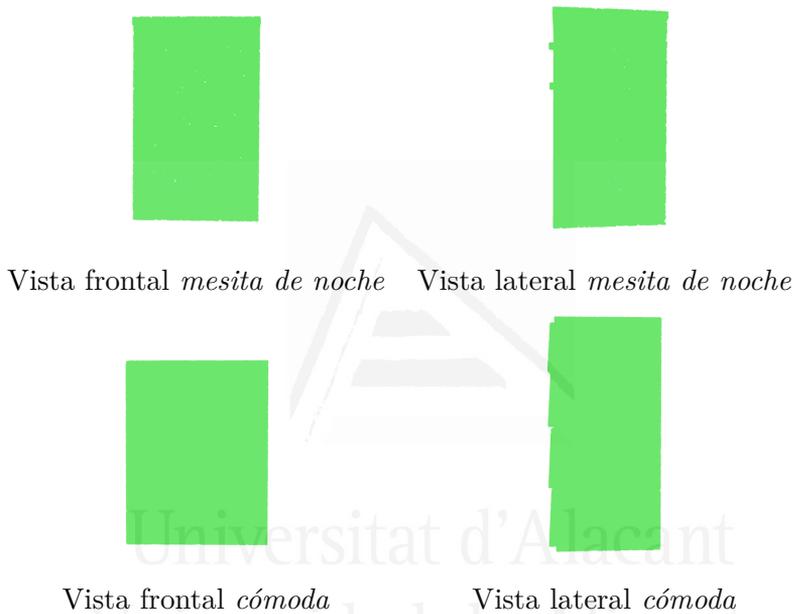


Figura A.18: Comparativa entre los casos de mesita de noche y cómoda en vista frontal y lateral

ModelNet40 ofrece un conjunto de más de 11.000 modelos CAD de 40 categorías diferentes alineados manualmente y divididos en conjunto de entrenamiento y de prueba. Al igual que en el conjunto de datos *ModelNet10*, convertimos estos modelos en nubes PCD. Esta versión del conjunto de datos también muestra una distribución muy desequilibrada de los conjuntos de entrenamiento y de prueba.

El problema referido a *ModelNet10* ocurre también con este conjunto de datos, como se muestra en las Figuras A.19 y A.20, ya que es de la misma naturaleza. Además de las anteriores, veremos frecuentes confusiones entre las clases *puerta* y *cortina*, porque ambas son rectángulos lisos,

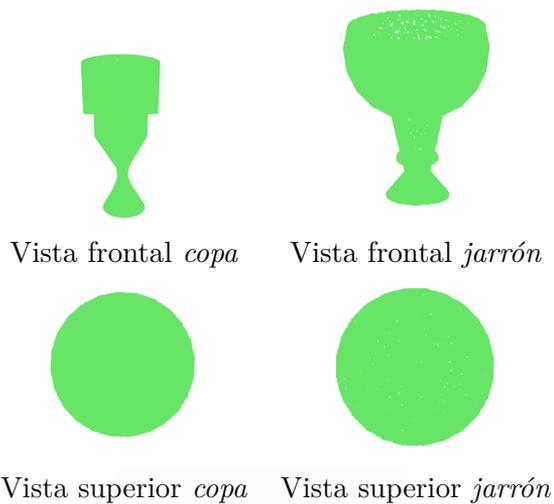


Figura A.19: Comparativa entre instancias de *copa* y *jarrón* en vista frontal y superior.

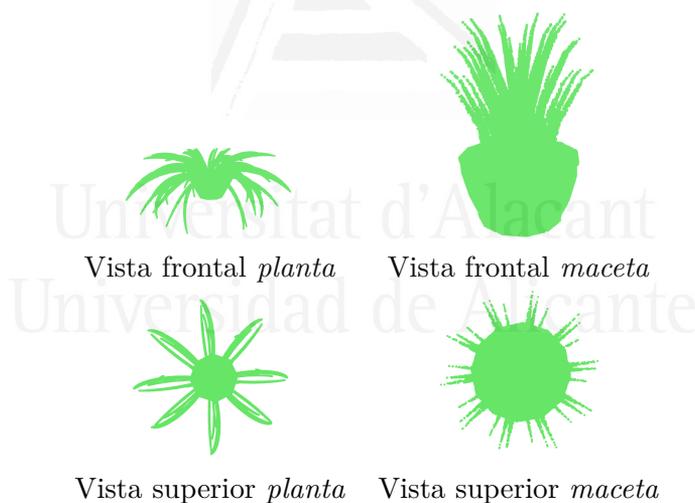


Figura A.20: Comparativa entre instancias de *planta* y *maceta* en vista frontal y superior.

más altos que anchos y poco profundos. Algo parecido ocurre entre *banco* y *escritorio*, porque la mayoría de sus instancias están formadas por una gran base rectangular con cuatro patas. Sin embargo, las principales confusiones se dan entre las clases *maceta*, *planta*, *jarrón* y *copa*. Podemos encontrar muchas plantas en maceta en las instancias *maceta* y *planta* in-

distintamente, por lo que se espera que la clasificación sea algo arbitraria en este caso. Por otro lado, las categorías *maceta*, *jarrón* y *copa* tienen instancias muy similares que sólo se diferencian en la escala, el color y el uso, características que no se suelen tener en cuenta en los algoritmos de reconocimiento de objetos 3D.

A.2.11 Conjuntos de datos para el reconocimiento de acciones

Aunque en esta tesis no se realiza el reconocimiento de acciones en uso, se ha realizado un análisis de la idoneidad de los ejercicios de rehabilitación física comparando la similitud entre la secuencia de poses del modelo, realizada por un entrenador o terapeuta, y la realizada por los pacientes.

Para ello, necesitamos datos etiquetados de acciones y poses para poder evaluar la eficacia de las métricas de comparación propuestas en nuestro trabajo. En el estado del arte existen diferentes conjuntos de datos para la realización de esta tarea.

- Moments in Time [Monfort et al., 2019] es un conjunto de datos a gran escala para el reconocimiento de acciones y eventos en vídeos. Incluye una colección de un millón de vídeos de 3 segundos etiquetados de personas, animales, objetos y fenómenos naturales.
- El conjunto de datos AVA [Gu et al., 2018] incluye la anotación de 80 acciones atómicas, en lugar de acciones compuestas, distribuidas en 350.000 vídeos, anotadas con su correspondiente ubicación y fecha de grabación, para un total de 1.600.000 acciones etiquetadas.
- Kinetics [Kay et al., 2017] es un conjunto de datos a gran escala y de alta calidad de URL de vídeos de YouTube que incluyen una amplia gama de acciones centradas en el ser humano. Consta de aproximadamente 300.000 clips de vídeo y cubre 400 clases de acciones humanas con al menos 400 clips de vídeo para cada clase de acción. Cada clip dura unos 10 segundos y está etiquetado con una sola clase. Todos los clips se han sometido a múltiples rondas de anotación humana y cada clip procede de un único vídeo de YouTube. Las acciones abarcan una

amplia gama de clases que incluyen interacciones hombre-objeto, como tocar instrumentos, así como interacciones hombre-hombre, como estrechar la mano y abrazar.

Debido a las necesidades específicas de nuestro trabajo, utilizaremos el conjunto de datos KARD [Gaglio et al., 2015], ya que nos proporciona los esqueletos de las personas y nos permite aplicar directamente nuestra métrica de evaluación de similitud entre poses.

A.2.11.1 KARD

KARD [Gaglio et al., 2015] contiene 18 actividades diferentes de corta duración, resumidas en la Tabla A.1, que en general coinciden con las características de algunos ejercicios de rehabilitación. Se trata de 10 individuos que repiten 3 veces cada actividad. El conjunto de datos proporciona el vídeo de cada persona realizando una actividad y la posición de las articulaciones del cuerpo en coordenadas 2D y 3D, tal y como las captó un dispositivo Kinect. Además, proporciona los correspondientes mapas de profundidad. En la Figura A.21 se muestran ejemplos de imágenes de este conjunto de datos.

Cuadro A.1: Actividades incluidas en el KARD [Gaglio et al., 2015].

ID	Actividad	ID	Actividad	ID	Actividad
1	Agitar brazos horizontalmente	7	Dibujar una cruz	13	Aplausos
2	Saludo con el brazo en alto	8	Tirar papel	14	Caminar
3	Saludo con las dos manos	9	Patada hacia delante	15	Llamada de teléfono
4	Atrapar la gorra	10	Patada lateral	16	Beber
5	Lanzar alto	11	Coger el paraguas	17	Sentarse
6	Dibujar una X	12	Doblarse	18	Ponerse de pie



Figura A.21: Ejemplos de imágenes de KARD. Obtenidos de [Gaglio et al., 2015]

A.3 Propuesta y objetivos

El nexo común de esta tesis es la investigación en el campo de la robótica social. Para ello, se han dedicado grandes esfuerzos a las tareas de reconocimiento de objetos 3D y mapeo del entorno, ya que son dos campos de vital importancia para la interacción del robot con un entorno doméstico. También se han llevado a cabo investigaciones adicionales para dotar al robot de la capacidad de reconocer a las personas, y se ha desarrollado un sistema de apoyo a la rehabilitación que puede incorporarse al robot y que puede ayudar a la adherencia al tratamiento.

En concreto, los objetivos de este trabajo de investigación son los siguientes:

- Explorar diferentes alternativas para el reconocimiento de objetos en 3D.
- Encontrar nuevas formas de mapear el entorno, incluyendo información útil sobre los objetos del entorno.
- Permitir que el robot realice un trato personalizado con las personas con las que interactúa y les ayude en tareas de rehabilitación.

A.4 Estructura de la tesis

De acuerdo con los temas presentados en la introducción, la estructura de la tesis es la siguiente.

En el capítulo 2, **Estado del arte**, se recogen los principales trabajos del estado del arte sobre reconocimiento de objetos 3D y mapeo del entorno. Además, se incluye información importante relacionada con la robótica social, que se ampliará más adelante en la tesis.

En el capítulo 3, **Reconocimiento de objetos 3D**, se exploran diferentes enfoques para el reconocimiento de objetos 3D, que son alternativas novedosas a los métodos actuales del estado del arte. Se utilizarán superficies libres y conceptos matemáticos como los fractales para obtener descriptores 3D.

En el capítulo 4, **Mapeado del entorno**, se exploran diferentes soluciones para dotar a un robot social de conocimientos sobre el entorno que le rodea. Estos enfoques incluyen un mapeo del entorno que incluye información sobre los objetos presentes en la escena, un planificador de rutas que permite optimizar los recorridos entre las diferentes estancias de la casa, y un mapeo que permite evaluar y detectar los posibles riesgos a los que puede estar expuesta una persona dentro de una casa. Este capítulo también propone una solución al problema del sensor de profundidad del robot Pepper, basada en la fusión sensorial y la inferencia de profundidad a partir de imágenes en color.

En el capítulo 5, **Investigaciones aplicadas a la robótica social**, se incluyen aportaciones relacionadas con la robótica social que no encajan directamente en los apartados anteriores. En primer lugar, hacemos una amplia revisión de la robótica social aplicada al campo de la atención a personas mayores y a la terapia con personas con autismo. En segundo lugar, presentamos un sistema de realidad aumentada que permite realizar terapias de rehabilitación física en casa y que proporciona a los profesionales un feedback que les permite evaluar los resultados. Por último, presentamos un módulo de reconocimiento humano capaz de aprender identidades en tiempo real, que permite personalizar la interacción del robot con los humanos.

En el capítulo 6, **Conclusiones**, se exponen las conclusiones y consideraciones finales. Además, se presentan las aportaciones al tema y se enumeran las publicaciones derivadas de este trabajo. Para finalizar el capítulo, también se muestran las direcciones futuras de la investigación realizada.



Universitat d'Alacant
Universidad de Alicante



Universitat d'Alacant
Universidad de Alicante

Conclusiones

Las conclusiones de este trabajo se recogen en este capítulo. El apartado B.1 contiene las conclusiones generales sobre el trabajo realizado en la tesis. Las principales aportaciones de esta tesis se recogen en el apartado B.2. Las publicaciones que se han realizado durante la elaboración de la tesis se recogen en el apartado B.3. Por último, en el apartado B.4 se exponen las futuras líneas de investigación para mejorar y ampliar el trabajo realizado en esta tesis.

B.1 Conclusiones

En esta tesis se han propuesto diferentes métodos alternativos para la realización del reconocimiento de objetos 3D.

En el caso de NurbsNet, se ha introducido por primera vez la similitud entre formas libres, nurbs en este caso, para determinar la clasificación global de una nube de puntos que representa un objeto. Para ello, ha sido necesario investigar una métrica de distancia que permita comparar adecuadamente dos superficies en términos de similitud de formas. A pesar de no superar el estado del arte, este trabajo abre nuevas vías de investigación inéditas para un reconocimiento de objetos más orgánico y fácilmente comprensible para el ser humano.

En el caso de VFD, se ha utilizado por primera vez el concepto matemático de la dimensión fractal para el reconocimiento de objetos tridimensionales. Los resultados de este método son prometedores e invitan a seguir

investigando sobre los fractales aplicados al reconocimiento de objetos y a otros campos del procesamiento de datos 3D.

Cabe destacar que los métodos utilizados utilizan la nube de puntos como entrada, lo cual es una ventaja ya que es la representación elegida por la gran mayoría de los sensores tridimensionales.

En cuanto al mapeo del entorno, hemos propuesto el desarrollo de un sistema de mapeo automático del entorno en 3D en el que no sólo se incluyen los puntos directamente en su ubicación correspondiente, sino que se ha realizado un procesamiento adicional para dar significado a estos puntos. En nuestra investigación hemos dotado a estos mapas del entorno del reconocimiento y segmentación de los objetos presentes en la escena, así como de la detección de zonas potencialmente peligrosas para las personas.

Asimismo, hemos propuesto un sistema de planificación que permite calcular rutas entre habitaciones utilizando un mapa semántico del entorno, y devuelve la ruta más óptima entre habitaciones teniendo en cuenta criterios de conectividad (pasillos y puertas abiertas) y coste (distancia, tiempo u otras circunstancias). Este sistema permite la navegación de un robot social entre habitaciones.

Además, hemos propuesto un método para fusionar datos tridimensionales de varias fuentes, que permite aprovechar las ventajas de cada una de ellas y superar sus desventajas.

En cuanto a la robótica, hemos realizado un estudio sobre el estado de la robótica social en relación con el cuidado de los ancianos, el envejecimiento activo y las terapias con autistas. De este modo, hemos podido saber dónde estamos y proponer soluciones a los problemas existentes.

Basándonos en esta investigación previa, hemos propuesto un sistema de realidad aumentada que permite ayudar en las tareas de rehabilitación física dentro de un hogar, y ser evaluado cuantitativamente con nuestras métricas, para que el terapeuta pueda conocer la evolución de la terapia.

Por último, hemos presentado un módulo de reconocimiento humano capaz de aprender nuevas identidades en tiempo real, lo que permitiría a los robots tratar a sus interlocutores de forma personalizada.

B.2 Aportaciones de la tesis

Las aportaciones realizadas en esta tesis son las siguientes:

- *NurbsNet* y *Voxelized Fractal Descriptor* son dos arquitecturas para el reconocimiento de objetos 3D a partir de nubes de puntos utilizando superficies de forma libre y dimensión fractal respectivamente.
- Mapeado automático en 3D del entorno que incluye información sobre los objetos del entorno y la delimitación de las zonas peligrosas.
- Un sistema de planificación de rutas basado en la conectividad entre habitaciones dentro de un mapa semántico, teniendo en cuenta el coste (tiempo u otras circunstancias) del desplazamiento entre habitaciones.
- Un método para combinar la información de un sensor de profundidad con las predicciones de profundidad de las imágenes monoculares, aprovechando las ventajas de cada uno y superando sus desventajas. De este modo, se han corregido los problemas inherentes al sensor de profundidad del robot Pepper.
- Una revisión del estado actual de la robótica social centrada en el campo de la atención a las personas mayores y a las personas con autismo, que da una idea del estado actual de las terapias de rehabilitación asistidas por robots.
- Un método para la evaluación de las posturas y los ejercicios de rehabilitación para la fisioterapia dentro del entorno doméstico del paciente.
- Un módulo de reidentificación de personas basado en la apariencia, integrable en un robot social, capaz de aprender nuevas identidades y realizar inferencias en tiempo real.

B.3 Publicaciones

Durante esta tesis, se han publicado 7 trabajos en revistas de alto impacto, calificadas por el Journal Citation Reports (JCR). Además, se han presentado 3 contribuciones en congresos internacionales.

En concreto, se han publicado los siguientes artículos como resultado de la investigación realizada durante esta tesis.

Artículos publicados en revistas científicas:

- Cruz, E., Escalona, F., Bauer, Z., Cazorla, M., García-Rodríguez, J., Martínez-Martin, E., and Gomez-Donoso, F. (2018). Geoffrey: an automated schedule system on a social robot for the intellectually challenged. *Computational intelligence and neuroscience*, 2018. [Cruz et al., 2018a] **(Capítulo 4.4)**
- Gomez-Donoso, F., Escalona, F., Rivas, F. M., Cañas, J. M., and Cazorla, M. (2019). Enhancing the ambient assisted living capabilities with a mobile robot. *Computational intelligence and neuroscience*, 2019. [Gomez-Donoso et al., 2019] **(Capítulo 4.3)**
- Escalona, F., Martínez-Martin, E., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). EVA: EVALuating at-home rehabilitation exercises using augmented reality and low-cost sensors. *Virtual Reality*, 1-15. [Escalona et al., 2019] **(Capítulo 5.3)**
- Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). Refining the Fusion of Pepper Robot and Estimated Depth Maps Method for Improved 3D Perception. *IEEE Access*, 7, 185076-185085. [Bauer et al., 2019] **(Capítulo 4.5)**
- Martínez-Martin, E., Escalona, F., and Cazorla, M. (2020). Socially assistive robots for older adults and people with autism: An overview. *Electronics*, 9(2), 367. [Martínez-Martin et al., 2020] **(Capítulo 5.2)**
- Martín-Rico, F., Gomez-Donoso, F., Escalona, F., García-Rodríguez, J., and Cazorla, M. (2020). Semantic visual recognition in a cognitive architecture for social robots. *Integrated Computer-Aided Engineering*, (Preprint), 1-16. [Martín-Rico et al., 2020] **(Capítulo 5.4)**

- Domenech, J. F., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2020). A Voxelized Fractal Descriptor for 3D Object Recognition. *IEEE Access*, 8, 161958-161968. [Domenech et al., 2020] **(Capítulo 3.3)**

Conferencias internacionales:

- Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2017, November). 3D Object Mapping Using a Labelling System. In *Iberian Robotics conference* (pp. 579-590). Springer, Cham. [Escalona et al., 2017a] **(Capítulo 4.2)**
- Martin-Rico, F., Gomez-Donoso, F., Escalona, F., Cazorla, M., and Garcia-Rodriguez, J. (2019, June). Artificial Semantic Memory with Autonomous Learning Applied to Social Robots. In *International Work-Conference on the Interplay Between Natural and Artificial Computation* (pp. 401-411). Springer, Cham. [Martin-Rico et al., 2019] **(Capítulo 5.4)**
- Escalona, F., Viejo, D., Fisher, R. B., and Cazorla, M. (2020, July). NurbsNet: A Nurbs approach for 3d object recognition. In *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-7). IEEE. [Escalona et al., 2020] **Seleccionado como finalista (5 candidatos) para el premio IJCNN-IEEE 2020 Best Student Paper Award (Capítulo 3.2)**

Otros trabajos a los que se ha contribuido en el transcurso de la tesis pero cuyo contenido no se ha incluido en la misma:

- Escalona, F., Rodriguez, A., Gomez-Donoso, F., Martinez-Gomez, J., and Cazorla, M. (2017). 3D object detection with deep learning. *Journal of Physical Agents*, 8(1), 3-10. [Escalona et al., 2017c]
- Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2018, November). Improving the 3D perception of the pepper robot using depth prediction from monocular frames. In *Workshop of Physical Agents* (pp. 132-146). Springer, Cham. [Bauer et al., 2018]

- Garcia-Rodriguez, J., Gomez-Donoso, F., Oprea, S., Garcia-Garcia, A., Cazorla, M., Orts-Escolano, S., ... and Rivas-Montero, F. (2020). COMBAHO: A deep learning system for integrating brain injury patients in society. *Pattern Recognition Letters*, 137, 80-90. [Garcia-Rodriguez et al., 2020]
- Escalona, F., Gomez-Donoso, F., Viejo, D., Orts-Escolano, S., and Cazorla, M. (2017). PRACTICAL CLASSES IN COMPUTER PROGRAMMING FOR A ROBOTICS ENGINEER. In *INTED2017 Proceedings* (pp. 2486-2490). IATED. [Escalona et al., 2017b]
- Torres-Camara, J. M., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2019, November). Map Slammer: Densifying Scattered KSLAM 3D Maps with Estimated Depth. In *Iberian Robotics conference* (pp. 563-574). Springer, Cham. [Torres-Camara et al., 2019]
- Gomez-Donoso, F., Escalona, F., and Cazorla, M. (2020). Par3dnet: Using 3dcnns for object recognition on tridimensional partial views. *Applied Sciences*, 10(10), 3409. [Gomez-Donoso et al., 2020b]
- Gomez-Donoso, F., Escalona, F., Nasri, N., and Cazorla, M. (2021). A Hand Motor Skills Rehabilitation for the Injured Implemented on a Social Robot. *Applied Sciences*, 11(7), 2943. [Gomez-Donoso et al., 2021a]
- Gomez-Donoso, F., Escalona, F., Bañuls, A., Abellan, D., and Cazorla, M. (2020, November). Monocular 3D Hand Pose Estimation for Teleoperating Low-Cost Actuators. In *Workshop of Physical Agents* (pp. 345-359). Springer, Cham. [Gomez-Donoso et al., 2020a]
- Gomez-Donoso, F., Escalona, F., Pérez-Esteve, F., and Cazorla, M. (2021). Accurate Multilevel Classification for Wildlife Images. *Computational Intelligence and Neuroscience*, 2021. [Gomez-Donoso et al., 2021b]

B.4 Trabajos futuros

En cuanto al reconocimiento de objetos en 3D, está previsto mejorar los dos enfoques que hemos propuesto.

Para el reconocimiento mediante formas libres, estamos investigando cuál es la mejor manera de incorporar la actualización y optimización de las superficies incrustadas en la capa de Nurbs para obtener automáticamente aquellas superficies que mejor diferencian los objetos, de forma análoga a como los filtros de convolución aprenden sus pesos para destacar las características que mejor representan el conjunto de datos, lo que puede permitirnos tanto mejorar los resultados de la clasificación como comprender mejor el proceso de toma de decisiones de la red.

En cuanto al reconocimiento de objetos mediante fractales, planeamos crear un descriptor local basado en los mismos fundamentos que este descriptor, de forma que seamos capaces de describir de forma única cada punto de una nube de puntos, lo que nos permitiría hacer un reconocimiento de objetos mediante modelos incompletos.

En cuanto al mapeo del entorno, se plantea modelar los obstáculos teniendo en cuenta que el entorno puede ser cambiante, tratando de separar las estructuras fijas de cada habitación (paredes, suelo) de otros elementos que pueden estar sujetos a cambios, como sillas, mesas o estanterías. De este modo, el robot detectaría los cambios en el entorno y actualizaría adecuadamente la información tanto de la estructura de la escena como de los objetos presentes en el entorno.

En cuanto al sistema de reidentificación de personas, nuestro objetivo es incorporar el reconocimiento facial siguiendo una estrategia similar a la seguida en el reconocimiento corporal implementado, de forma que el robot pueda seguir reconociendo convenientemente a las personas aunque éstas realicen cambios en su vestimenta.



Universitat d'Alacant
Universidad de Alicante

Bibliography

- [ten,] TensorFlow: Large-scale machine learning on heterogeneous systems. <https://www.tensorflow.org/>. 15, 227
- [Abdallah and Fayyoubi, 2016] Abdallah, E. E. and Fayyoubi, E. (2016). Assistive technology for deaf people based on android platform. *Procedia Computer Science*, 94:295–301. 150
- [Ahmadlou and Adeli, 2010] Ahmadlou, M. and Adeli, H. (2010). Enhanced probabilistic neural network with local decision circles: A robust classifier. *Integr. Comput.-Aided Eng.*, 17(3):197–210. 202
- [Ahmed et al., 2015] Ahmed, E., Jones, M. J., and Marks, T. K. (2015). An improved deep learning architecture for person re-identification. In *CVPR*, pages 3908–3916. IEEE Computer Society. 53
- [Ahonen et al., 2004] Ahonen, T., Hadid, A., and Pietikäinen, M. (2004). Face recognition with local binary patterns. In *European conference on computer vision*, pages 469–481. Springer. 162
- [Al-Issa et al., 2013] Al-Issa, H., Regenbrecht, H., and Hale, L. (2013). Augmented reality applications in rehabilitation to improve physical outcomes. *Physical Therapy Reviews*, 17(1):16–28. 50
- [Aldebaran-Robotics, 2014] Aldebaran-Robotics (2014). *Pepper Technical Docs*. http://doc.aldebaran.com/2-5/family/pepper_technical/index_pep.html (Accessed: 2018-06-29). 5, 6, 7, 8, 9, 10, 218, 219, 220, 221, 222

- [Aldoma et al., 2012a] Aldoma, A., Tombari, F., Di Stefano, L., and Vincze, M. (2012a). A global hypotheses verification method for 3d object recognition. In *European conference on computer vision*, pages 511–524. Springer. 33
- [Aldoma et al., 2012b] Aldoma, A., Tombari, F., Rusu, R. B., and Vincze, M. (2012b). Our-cvfh-oriented, unique and repeatable clustered view-point feature histogram for object recognition and 6dof pose estimation. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pages 113–122. Springer. 34
- [Aldoma et al., 2011] Aldoma, A., Vincze, M., Blodow, N., Gossow, D., Gedikli, S., Rusu, R. B., and Bradski, G. (2011). Cad-model recognition and 6dof pose estimation using 3d cues. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*, pages 585–592. IEEE. 34
- [Alhamzi et al., 2015] Alhamzi, K., Elmogy, M., and Barakat, S. (2015). 3d object recognition based on local and global features using point cloud library. *International Journal of Advancements in Computing Technology*, 7(3):43. 33
- [Amit, 2002] Amit, Y. (2002). *2D object detection and recognition: Models, algorithms, and networks*. MIT Press. 26
- [Amodei et al., 2016] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. (2016). Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. 163
- [Andrade-Cetto and Torras, 2006] Andrade-Cetto, J. and Torras, C. (2006). Paco-plus: Perception, action and cognition through learning of object-action complexes. 151
- [Annoy, 2018] Annoy (2018). <https://github.com/spotify/annoy>. Online; accessed 11th October 2018. 122

- [Armeni et al., 2017] Armeni, I., Sax, A., Zamir, A. R., and Savarese, S. (2017). Joint 2D-3D-Semantic Data for Indoor Scene Understanding. *ArXiv e-prints*. 17, 229
- [Askari, 2018] Askari, F. (2018). Studying facial expression recognition and imitation ability of children with autism spectrum disorder in interaction with a social robot. Technical report. 160
- [Askari et al., 2018] Askari, F., Feng, H., Sweeny, T. D., and Mahoor, M. H. (2018). A pilot study on facial expression recognition ability of autistic children using ryan, a rear-projected humanoid robot. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 790–795. IEEE. 160
- [Atapour-Abarghouei and Breckon, 2018] Atapour-Abarghouei, A. and Breckon, T. P. (2018). Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 137
- [Aulinas et al., 2008] Aulinas, J., Petillot, Y. R., Salvi, J., and Lladó, X. (2008). The slam problem: a survey. In *CCIA*, pages 363–371. Citeseer. 39
- [Aung and Al-Jumaily, 2014] Aung, Y. M. and Al-Jumaily, A. (2014). Augmented reality-based rehabio system for shoulder rehabilitation. *International Journal on Mechatronics and Automation*, 4(1):52–62. 50
- [Aung et al., 2014] Aung, Y. M., Al-Jumaily, A., and Anam, K. (2014). A novel upper limb rehabilitation system with self-driven virtual arm illusion. In *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3614–3617, Chicago, IL, USA. 50
- [Azmin et al., 2016] Azmin, A. F., Shamsuddin, S., and Yussof, H. (2016). Hri observation with my keepon robot using kansei engineering approach. In *2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation (ROMA)*, pages 1–6. IEEE. 159

- [Azuar et al., 2019] Azuar, D., Gallud, G., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2019). A story-telling social robot with emotion recognition capabilities for the intellectually challenged. In *Iberian Robotics conference*, pages 599–609. Springer. 159, 163
- [Bae and Yoon, 2014] Bae, S.-H. and Yoon, K.-J. (2014). Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1218–1225. 188
- [Baluja and Pomerleau, 1994] Baluja, S. and Pomerleau, D. (1994). Non-intrusive gaze tracking using artificial neural networks. In *Advances in Neural Information Processing Systems*, pages 753–760. 161
- [Barcala et al., 2013] Barcala, L., Grecco, L. A. C., Colella, F., Lucareli, P. R. G., Salgado, A. S. I., and Oliveira, C. S. (2013). Visual biofeedback balance training using wii fit after stroke: A randomized controlled trial. *Journal of Physical Therapy Science*, 25(8):1027–1032. 50
- [Bauer et al., 2018] Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2018). Improving the 3d perception of the pepper robot using depth prediction from monocular frames. In *Workshop of Physical Agents*, pages 132–146. Springer, Cham. 207, 243
- [Bauer et al., 2019] Bauer, Z., Escalona, F., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). Refining the fusion of pepper robot and estimated depth maps method for improved 3d perception. *IEEE Access*, 7:185076–185085. 206, 242
- [Bayramoglu and Alatan, 2010] Bayramoglu, N. and Alatan, A. A. (2010). Shape index sift: Range image recognition using local features. In *2010 20th International Conference on Pattern Recognition*, pages 352–355. IEEE. 33
- [Begum et al., 2016] Begum, M., Serna, R. W., and Yanco, H. A. (2016). Are robots ready to deliver autism interventions? a comprehensive review. *International Journal of Social Robotics*, 8(2):157–181. 157

- [Bellhumeur et al., 1997] Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 19(7):711–720. 162
- [Bengio, 2009] Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127. 90
- [Berndt and Clifford, 1994] Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA. 173
- [Bhattacharya and Gavrilova, 2008] Bhattacharya, P. and Gavrilova, M. (2008). Roadmap-based path planning - using the voronoi diagram for a clearance-based shortest path. *IEEE Robot. Automat. Mag.*, 15(2):58–66. 38
- [Bhowmick and Hazarika, 2017] Bhowmick, A. and Hazarika, S. M. (2017). An insight into assistive technology for the visually impaired and blind people: state-of-the-art and future trends. *Journal on Multimodal User Interfaces*, 11(2):149–172. 150
- [Billard, 2003] Billard, A. (2003). Robota: Clever toy and educational tool. *Robotics and Autonomous Systems*, 42(3-4):259–269. 160
- [Billard et al., 2006] Billard, A., Robins, B., Dautenhahn, K., and Nadel, J. (2006). Building robota, a mini-humanoid robot for the rehabilitation of children with autism. *The RESNA Assistive Technology Journal*, 19(ARTICLE). 160
- [Blake and Isard, 1998] Blake, A. and Isard, M. (1998). *Active contours: the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion*. Springer Science & Business Media. 58
- [Bochkovskiy et al., 2020] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*. 15, 227

- [Booij et al., 2007] Booij, O., Terwijn, B., Zivkovic, Z., and Kröse, B. (2007). Navigation using an appearance based topological map. In *Int. Conf. on Robotics and Automation*. IEEE. 38
- [Breunig et al., 2000] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM. 199
- [British National Health Security (NHS), 2018] British National Health Security (NHS) (2018). Exercises for older people. https://www.nhs.uk/Tools/Documents/NHS_ExercisesForOlderPeople.pdf. 180
- [Brock et al., 2016] Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2016). Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*. 37
- [Brown, 2018] Brown, A. S. (2018). Face to face with autism. *Mechanical Engineering Magazine Select Articles*, 140(02):35–39. 159
- [Bruno et al., 2008] Bruno, O. M., de Oliveira Plotze, R., Falvo, M., and de Castro, M. (2008). Fractal dimension applied to plant identification. *Information Sciences*, 178(12):2722–2733. 37
- [Burkhardt et al., 2019] Burkhardt, F., Saponja, M., Sessner, J., and Weiss, B. (2019). How should pepper sound-preliminary investigations on robot vocalizations. *Studenttexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2019*, pages 103–110. 159
- [Cabibihan et al., 2013] Cabibihan, J.-J., Javed, H., Ang, M., and Aljunied, S. M. (2013). Why robots? a survey on the roles and benefits of social robots in the therapy of children with autism. *International journal of social robotics*, 5(4):593–618. 158
- [Calli et al., 2015] Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., and Dollar, A. M. (2015). The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE. 16, 228

- [Cao et al., 2017] Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *Computer Vision and Pattern Recognition (CVPR)*. 119, 170
- [Cazorla et al., 2016] Cazorla, M., Viejo, D., and Gómez, F. (2016). Mapeado 3d de interiores. Departamento de Ciencia de la Computación e Inteligencia Artificial. Universidad de Alicante https://moodle2015-16.ua.es/moodle/pluginfile.php/105464/mod_page/content/11/Practica2VAR2016.pdf. 13, 225
- [Chakraborty and Canas, 2016] Chakraborty, S. and Canas, J. M. (2016). Making compatible two robotic middlewares: Ros and jderobot. In *Proceedings of XVII Workshop on Physical Agents (WAF-2016)*, pages 147–154, Málaga, Spain. 113
- [Chan et al., 2016] Chan, W., Jaitly, N., Le, Q., and Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964. IEEE. 163
- [Chandola et al., 2009] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15. 190
- [Charalampous and Gasteratos, 2016] Charalampous, K. and Gasteratos, A. (2016). On-line deep learning method for action recognition. *Pattern Analysis and Applications*, 19(2):337–354. 188
- [Chen et al., 2001] Chen, Y., Zhou, X. S., and Huang, T. S. (2001). One-class svm for learning in image retrieval. In *ICIP (1)*, pages 34–37. Citeseer. 199
- [Chen et al., 2017] Chen, Y., Zhu, X., and Gong, S. (2017). Person re-identification by deep learning multi-scale representations. In *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017*, pages 2590–2600. 53

- [Cheng et al., 2015] Cheng, M.-M., Mitra, N. J., Huang, X., Torr, P. H., and Hu, S.-M. (2015). Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582. 30
- [Chevalier et al., 2016] Chevalier, P., Isableu, B., Martin, J.-C., and Tapus, A. (2016). Individuals with autism: Analysis of the first interaction with nao robot based on their proprioceptive and kinematic profiles. In *Advances in robot design and intelligent control*, pages 225–233. Springer. 159
- [Choi et al., 2016] Choi, S., Zhou, Q.-Y., Miller, S., and Koltun, V. (2016). A large dataset of object scans. *arXiv preprint arXiv:1602.02481*. 16, 229
- [Chollet et al., 2015] Chollet, F. et al. (2015). Keras. <https://keras.io>. 15, 227
- [Chrschn, 2019] Chrschn (2019). Nurbs surface. <https://es.wikipedia.org/wiki/NURBS>. Accessed: 2019-12-05. 57
- [Cireşan et al., 2010] Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220. 32
- [Coşar et al., 2020] Coşar, S., Fernandez-Carmona, M., Agrigoroaie, R., Pages, J., Ferland, F., Zhao, F., Yue, S., Bellotto, N., and Tapus, A. (2020). Enrichme: Perception and interaction of an assistive robot for the elderly at home. *International Journal of Social Robotics*, 12(3):779–805. 151
- [Costa et al., 2016] Costa, A., Julián, V., and Novais, P. (2016). Advances and trends for the development of ambient-assisted living platforms. *Expert Systems*, 34(2):e12163. 151
- [Costa et al., 2018a] Costa, A., Martinez-Martin, E., Cazorla, M., and Julian, V. (2018a). Pharos - physical assistant robot system. *Sensors*. 48, 49, 153

- [Costa et al., 2017] Costa, A., Novais, P., and Julian, V. (2017). A survey of cognitive assistants. In *Intelligent Systems Reference Library*, pages 3–16. Springer International Publishing. 151
- [Costa et al., 2018b] Costa, A., Novais, P., Julian, V., and Nalepa, G. J. (2018b). Cognitive assistants. *International Journal of Human-Computer Studies*, 117:1–3. 151
- [Cruz et al., 2018a] Cruz, E., Escalona, F., Bauer, Z., Cazorla, M., García-Rodríguez, J., Martínez-Martin, E., Rangel, J. C., and Gomez-Donoso, F. (2018a). Geoffrey: an automated schedule system on a social robot for the intellectually challenged. *Computational intelligence and neuroscience*, 2018. 151, 206, 242
- [Cruz et al., 2018b] Cruz, E., Rangel, J. C., Gomez-Donoso, F., Bauer, Z., Cazorla, M., and García-Rodríguez, J. (2018b). Finding the place: how to train and use convolutional neural networks for a dynamically learning robot. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE. 120
- [Curtis et al., 2011] Curtis, A., Shim, J., Gargas, E., Srinivasan, A., and Howard, A. M. (2011). Dance dance pleo: developing a low-cost learning robotic dance therapy aid. In *Proceedings of the 10th International Conference on Interaction Design and Children*, pages 149–152. Citeseer. 159
- [Dai et al., 2017] Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. (2017). Scannet: Richly-annotated 3d reconstructions of indoor scenes. cite arxiv:1702.04405. 17, 229
- [Damm et al., 2013] Damm, O., Malchus, K., Jaecks, P., Krach, S., Paulus, F., Naber, M., Jansen, A., Kamp-Becker, I., Einhaeuser-Treyer, W., Stenneken, P., et al. (2013). Different gaze behavior in human-robot interaction in asperger’s syndrome: An eye-tracking study. In *2013 IEEE RO-MAN*, pages 368–369. IEEE. 161, 162
- [Dautenhahn and Billard, 2002] Dautenhahn, K. and Billard, A. (2002). Games children with autism can play with robots, a humanoid robotic

- doll. In *Universal access and assistive technology*, pages 179–190. Springer. 160
- [Dautenhahn et al., 2009] Dautenhahn, K., Nehaniv, C. L., Walters, M. L., Robins, B., Kose-Bagci, H., Mirza, N. A., and Blow, M. (2009). Kaspar—a minimally expressive humanoid robot for human–robot interaction research. *Applied Bionics and Biomechanics*, 6(3-4):369–397. 159
- [Dautenhahn and Werry, 2004] Dautenhahn, K. and Werry, I. (2004). Towards interactive robots in autism therapy: Background, motivation and challenges. *Pragmatics & Cognition*, 12(1):1–35. 158
- [Davydov and Lozynska, 2016] Davydov, M. and Lozynska, O. (2016). Linguistic models of assistive computer technologies for cognition and communication. In *2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT)*. IEEE. 150
- [de Assis et al., 2014] de Assis, G. A., Correa, A. G. D., Rodrigues Martins, M. B., Pedrozo, W. G., and de Deus Lopes, R. (2014). An augmented reality system for upper-limb post-stroke motor rehabilitation: a feasibility study. *Disability and Rehabilitation: Assistive Technology*, 11(16):521–526. 50
- [de Oliveira et al., 2016] de Oliveira, G. A. A., de Bettio, R. W., and Freire, A. P. (2016). Accessibility of the smart home for users with visual disabilities. In *Proceedings of the 15th Brazilian Symposium on Human Factors in Computer Systems - IHC '16*. ACM Press. 151
- [Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*. 33, 92, 192
- [Deng et al., 2019] Deng, J., Guo, J., Xue, N., and Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 162

- [Desai et al., 2016] Desai, K., Bahirat, K., Ramalingam, S., Prabhakaran, B., Annaswamy, T., and Makris, U. E. (2016). Augmented reality-based exergames for rehabilitation. In *7th International Conference on Multimedia Systems*, Klagenfurt, Austria. 51
- [Dickinson et al., 1992] Dickinson, S. J., Pentland, A., and Rosenfeld, A. (1992). 3-d shape recovery using distributed aspect matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):174–198. 28
- [Dickstein-Fischer et al., 2018] Dickstein-Fischer, L. A., Crone-Todd, D. E., Chapman, I. M., Fathima, A. T., and Fischer, G. S. (2018). Socially assistive robots: current status and future prospects for autism interventions. *Innovation and Entrepreneurship in Health*, 5:15. 157
- [Dickstein-Fischer et al., 2017] Dickstein-Fischer, L. A., Pereira, R. H., Gandomi, K. Y., Fathima, A. T., and Fischer, G. S. (2017). Interactive tracking for robot-assisted autism therapy. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 107–108. ACM. 159
- [Ding et al., 2014] Ding, J., Lim, Y., Solano, M., Shadle, K., Park, C., Lin, C., and Hu, J. (2014). Giving patients a lift - the robotic nursing assistant (rona). In *2014 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–5. 48
- [Domenech et al., 2020] Domenech, J. F., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2020). A voxelized fractal descriptor for 3d object recognition. *IEEE Access*, 8:161958–161968. 206, 243
- [dos Santos Mendesa et al., 2012] dos Santos Mendesa, F. A., Pompeua, J. E., Lobo, A. M., da Silva, K. G., de Paula Oliveira, T., Zomignani, A. P., and Piemonte, M. E. P. (2012). Motor learning, retention and transfer after virtual-reality-based training in parkinson’s disease – effect of motor and cognitive demands of games: a longitudinal, controlled clinical study. *Physiotherapy*, 98:217–223. 50

- [Du et al., 2007] Du, S., Zheng, N., Ying, S., You, Q., and Wu, Y. (2007). An extension of the icp algorithm considering scale factor. In *2007 IEEE International Conference on Image Processing*, volume 5, pages V–193. IEEE. 141
- [Duffy et al., 1999] Duffy, B. R., Rooney, C., O’Hare, G. M., and O’Donoghue, R. (1999). What is a social robot? In *10th Irish Conference on Artificial Intelligence & Cognitive Science, University College Cork, Ireland, 1-3 September, 1999*. 45
- [Duquette et al., 2008] Duquette, A., Michaud, F., and Mercier, H. (2008). Exploring the use of a mobile robot as an imitation agent with children with low-functioning autism. *Autonomous Robots*, 24(2):147–157. 160
- [Edelman and Bühlhoff, 1992] Edelman, S. and Bühlhoff, H. H. (1992). Orientation dependence in the recognition of familiar and novel views of three-dimensional objects. *Vision research*, 32(12):2385–2400. 28
- [English et al., 2017] English, B. A., Coates, A., and Howard, A. (2017). Recognition of gestural behaviors expressed by humanoid robotic platforms for teaching affect recognition to children with autism—a healthy subjects pilot study. In *International Conference on Social Robotics*, pages 567–576. Springer. 159
- [Escalona et al., 2017a] Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2017a). 3d object mapping using a labelling system. In *Iberian Robotics conference*, pages 579–590. Springer. 207, 243
- [Escalona et al., 2017b] Escalona, F., Gómez-Donoso, F., Viejo, D., Orts-Escolano, S., and Cazorla, M. (2017b). Practical classes in computer programming for a robotics engineer. In *INTED2017 Proceedings*, pages 2486–2490. IATED. 208, 244
- [Escalona et al., 2019] Escalona, F., Martinez-Martin, E., Cruz, E., Cazorla, M., and Gomez-Donoso, F. (2019). EVA: EVALuating at-home rehabilitation exercises using augmented reality and low-cost sensors. *Virtual Reality*. 151, 206, 242

- [Escalona et al., 2017c] Escalona, F., Rodríguez, A., Gómez-Donoso, F., Martínez-Gómez, J., and Cazorla, M. (2017c). 3d object detection with deep learning. *Journal of Physical Agents*, 8(1). 102, 207, 243
- [Escalona et al., 2020] Escalona, F., Viejo, D., Fisher, R. B., and Cazorla, M. (2020). Nurbsnet: A nurbs approach for 3d object recognition. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE. 207, 243
- [ETH-Zurich, 2014] ETH-Zurich (2014). *Automatic registration of partially overlapping terrestrial laser scanner point clouds*. http://www.prs.igp.ethz.ch/research/completed_projects/automatic_registration_of_point_clouds.html (Accessed: 2017-04-30). 42
- [EU project, 2013] EU project (2007–2013). Hobbit - the mutual care robot. <http://hobbit.acin.tuwien.ac.at/>. 154
- [EU project, 2020] EU project (2015-2020). Ramcip - robotic assistant for mci patients at home. <https://ramcip-project.eu>. Accessed on January 2020. 154
- [Falcone et al., 2003] Falcone, E., Gockley, R., Porter, E., and Nourbakhsh, I. (2003). The personal rover project:: The comprehensive design of a domestic personal robot. *Robotics and Autonomous Systems*, 42(3):245 – 258. Socially Interactive Robots. 46
- [Fasola and Mataric, 2010] Fasola, J. and Mataric, M. J. (2010). Robot exercise instructor: A socially assistive robot system to monitor and encourage physical exercise for the elderly. In *19th International Symposium in Robot and Human Interactive Communication*, pages 416–421. 47
- [Feil-Seifer and Mataric, 2005] Feil-Seifer, D. and Mataric, M. J. (2005). Defining socially assistive robotics. In *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pages 465–468. IEEE. 151

- [Firman, 2016] Firman, M. (2016). Rgbd datasets: Past, present and future. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 19–31. 16, 228
- [Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395. 45, 108, 135
- [Fox et al., 1999] Fox, D., Burgard, W., Dellaert, F., and Thrun, S. (1999). Monte carlo localization: Efficient position estimation for mobile robots. 126
- [Fung et al., 2012] Fung, V., Ho, A., Shaffer, J., Chung, E., and Gomez, M. (2012). Use of nintendo wii fit in the rehabilitation of outpatients following total knee replacement: a preliminary randomised controlled trial. *Physiotherapy*, 98:183–188. 50
- [Gaglio et al., 2015] Gaglio, S., Re, G. L., and Morana, M. (2015). Human activity recognition process using 3-d posture data. *IEEE Transactions on Human-Machine Systems*, 45(5):586–597. 20, 21, 181, 233, 234
- [Garcia-Garcia et al., 2016] Garcia-Garcia, A., Gomez-Donoso, F., Garcia-Rodriguez, J., Orts-Escolano, S., Cazorla, M., and Azorin-Lopez, J. (2016). Pointnet: A 3d convolutional neural network for real-time object class recognition. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1584. IEEE. 17, 36, 62, 83, 229
- [Garcia-Rodriguez et al., 2020] Garcia-Rodriguez, J., Gomez-Donoso, F., Oprea, S., Garcia-Garcia, A., Cazorla, M., Orts-Escolano, S., Bauer, Z., Castro-Vargas, J., Escalona, F., Ivorra-Piqueres, D., et al. (2020). Combaho: A deep learning system for integrating brain injury patients in society. *Pattern Recognition Letters*, 137:80–90. 3, 208, 215, 244
- [Gazzoni and Cerone, 2018] Gazzoni, M. and Cerone, G. L. (2018). Augmented reality system for muscle activity biofeedback. *Annals of Physical and Rehabilitation Medicine*, 61:e483–e484. 50

- [Geminiani et al., 2019] Geminiani, A., Santos, L., Casellato, C., Farabbi, A., Farella, N., Santos-Victor, J., Olivieri, I., and Pedrocchi, A. (2019). Design and validation of two embodied mirroring setups for interactive games with autistic children using the nao humanoid robot. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1641–1644. IEEE. 159
- [Giannopulu and Pradel, 2010] Giannopulu, I. and Pradel, G. (2010). Multimodal interactions in free game play of children with autism and a mobile toy robot. *NeuroRehabilitation*, 27(4):305–311. 160
- [Godard et al., 2017] Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279. 137
- [Goldsmith and LeBlanc, 2004] Goldsmith, T. R. and LeBlanc, L. A. (2004). Use of technology in interventions for children with autism. *Journal of Early and Intensive Behavior Intervention*, 1(2):166. 157
- [Goldstein and Dengel, 2012] Goldstein, M. and Dengel, A. (2012). Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: Poster and Demo Track*, pages 59–63. 199
- [Gómez et al., 2009] Gómez, C., Mediavilla, Á., Hornero, R., Abásolo, D., and Fernández, A. (2009). Use of the higuchi’s fractal dimension for the analysis of meg recordings from alzheimer’s disease patients. *Medical engineering & physics*, 31(3):306–313. 38
- [Gomez-Donoso et al., 2020a] Gomez-Donoso, F., Escalona, F., Bañuls, A., Abellan, D., and Cazorla, M. (2020a). Monocular 3d hand pose estimation for teleoperating low-cost actuators. In *Workshop of Physical Agents*, pages 345–359. Springer, Cham. 208, 244
- [Gomez-Donoso et al., 2020b] Gomez-Donoso, F., Escalona, F., and Cazorla, M. (2020b). Par3dnet: Using 3dcnns for object recognition on tridimensional partial views. *Applied Sciences*, 10(10):3409. 36, 208, 244

- [Gomez-Donoso et al., 2021a] Gomez-Donoso, F., Escalona, F., Nasri, N., and Cazorla, M. (2021a). A hand motor skills rehabilitation for the injured implemented on a social robot. *Applied Sciences*, 11(7):2943. 208, 244
- [Gomez-Donoso et al., 2021b] Gomez-Donoso, F., Escalona, F., Pérez-Esteve, F., and Cazorla, M. (2021b). Accurate multilevel classification for wildlife images. *Computational Intelligence and Neuroscience*, 2021. 208, 244
- [Gomez-Donoso et al., 2019] Gomez-Donoso, F., Escalona, F., Rivas, F. M., Cañas, J. M., and Cazorla, M. (2019). Enhancing the ambient assisted living capabilities with a mobile robot. *Computational intelligence and neuroscience*, 2019. 151, 206, 242
- [Gomez-Donoso et al., 2017a] Gomez-Donoso, F., Garcia-Garcia, A., Garcia-Rodriguez, J., Orts-Escolano, S., and Cazorla, M. (2017a). Lonchanet: A sliced-based cnn architecture for real-time 3d object recognition. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 412–418. IEEE. 36, 83
- [Gomez-Donoso et al., 2017b] Gomez-Donoso, F., Orts-Escolano, S., Garcia-Garcia, A., Garcia-Rodriguez, J., Castro-Vargas, J. A., Ovidiu-Oprea, S., and Cazorla, M. (2017b). A robotic platform for customized and interactive rehabilitation of persons with disabilities. *Pattern Recognition Letters*, 99:105 – 113. 49
- [Gong et al., 2014] Gong, S., Cristani, M., Yan, S., and Loy, C. C. (2014). *Person Re-Identification*. Springer Publishing Company, Incorporated. 52
- [Görer et al., 2016] Görer, B., Salah, A. A., and Akın, H. L. (2016). An autonomous robotic exercise tutor for elderly people. *Autonomous Robots*, 41(3):657–678. 153
- [Görer et al., 2017] Görer, B., Salah, A. A., and Akin, H. L. (2017). An autonomous robotic exercise tutor for elderly people. *Auton. Robots*, 41(3):657–678. 48

- [Gray and Tao, 2008] Gray, D. and Tao, H. (2008). Viewpoint invariant pedestrian recognition with an ensemble of localized features. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision – ECCV 2008*, pages 262–275, Berlin, Heidelberg. Springer Berlin Heidelberg. 52
- [Grisetti et al., 2007] Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46. 13, 14, 126, 225, 226
- [Gu et al., 2018] Gu, C., Sun, C., Ross, D. A., Vondrick, C., Pantofaru, C., Li, Y., Vijayanarasimhan, S., Toderici, G., Ricco, S., Sukthankar, R., et al. (2018). Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056. 20, 232
- [Guennebaud et al., 2010] Guennebaud, G., Jacob, B., et al. (2010). *Eigen v3*. <http://eigen.tuxfamily.org> (Accessed: 2017-05-02). 15, 227
- [Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the Fourth Alvey Vision Conference*, volume 15, pages 147–151. Citeseer. 29
- [He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385. 189
- [He et al., 2003] He, Z., Xu, X., and Deng, S. (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650. 199
- [Higuchi, 1988] Higuchi, T. (1988). Approach to an irregular time series on the basis of the fractal theory. *Physica D: Nonlinear Phenomena*, 31(2):277–283. 38
- [Hillman et al., 2002] Hillman, M., Hagan, K., Hagan, S., Jepson, J., and Orpwood, R. (2002). The weston wheelchair mounted assistive robot - the design story. *Robotica*, 20(2):125–132. 46

- [Hirzer et al., 2011] Hirzer, M., Beleznai, C., Roth, P. M., and Bischof, H. (2011). Person re-identification by descriptive and discriminative classification. In Heyden, A. and Kahl, F., editors, *Image Analysis*, pages 91–102, Berlin, Heidelberg. Springer Berlin Heidelberg. 53
- [Hocoma, 2018] Hocoma (2018). Lokomat. <https://esa.un.org/unpd/wpp/>. 49
- [Hsu et al., 2012] Hsu, P. E., Hsu, Y. L., Chang, K. W., and Geiser, C. (2012). Mobility assistance design of the intelligent robotic wheelchair. *International Journal of Advanced Robotic Systems*, 9(6):244. 46
- [Huete et al., 2012] Huete, A. J., Victores, J. G., Martinez, S., Gimenez, A., and Balaguer, C. (2012). Personal autonomy rehabilitation in home environments by a portable assistive robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):561–570. 47
- [Huijnen et al., 2016] Huijnen, C. A., Lexis, M. A., and de Witte, L. P. (2016). Matching robot kaspar to autism spectrum disorder (asd) therapy and educational goals. *International Journal of Social Robotics*, 8(4):445–455. 159
- [Huletski and Kartashov, 2016] Huletski, A. and Kartashov, D. (2016). A slam research framework for ros. In *Proceedings of the 12th Central and Eastern European Software Engineering Conference in Russia, CEE-SECR '16*, pages 12:1–12:6, New York, NY, USA. ACM. 111
- [Ikeda et al., 2005] Ikeda, S., Arai, F., Fukuda, T., Kim, E. H., Negoro, M., Irie, K., and Takahashi, I. (2005). In vitro patient-tailored anatomical model of cerebral artery for evaluating medical robots and systems for intravascular neurosurgery. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1558–1563. 46
- [Ikeda et al., 2006] Ikeda, S., Arai, F., Fukuda, T., Negoro, M., Irie, K., and Takahashi, I. (2006). Patient-specific neurovascular simulator for

- evaluating the performance of medical robots and instruments. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 625–630. 46
- [Indra, 2019] Indra (2019). Toyra. <https://www.tecnologiasaccesibles.com/es/proyectos/toyra>. 50
- [INF Robotics, 2019] INF Robotics (2019). Rudy. <http://infrobotics.com/>. Accessed on January 2020. 155
- [Ip et al., 2002] Ip, C. Y., Lapadat, D., Sieger, L., and Regli, W. C. (2002). Using shape distributions to compare solid models. In *Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 273–280. 34
- [Ishikawa, 2004] Ishikawa, T. (2004). Passive driver gaze tracking with active appearance models. Technical report. 161
- [Ismail et al., 2011] Ismail, L., Shamsuddin, S., Yussof, H., Hashim, H., Bahari, S., Jaafar, A., and Zahari, I. (2011). Face detection technique of humanoid robot nao for application in robotic assistive therapy. In *2011 IEEE International Conference on Control System, Computing and Engineering*, pages 517–521. IEEE. 163
- [Jack et al., 2001] Jack, D., Boian, R., Merians, A. S., Tremaine, M., Burdea, G. C., Adamovich, S. V., Recce, M., and Poizner, H. (2001). Virtual reality-enhanced stroke rehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 9(3):308–318. 49
- [Janssens et al., 2012] Janssens, J., Huszár, F., Postma, E., and van den Herik, H. (2012). Stochastic outlier selection. *tech. rep.* 199
- [Jelinek, 1997] Jelinek, F. (1997). *Statistical methods for speech recognition*. MIT press. 163
- [Jia et al., 2014] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proc. 22nd ACM International Conference on Multimedia*. 90

- [Johns et al., 2016] Johns, E., Leutenegger, S., and Davison, A. J. (2016). Pairwise decomposition of image sequences for active multi-view recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3813–3822. 36
- [Juang and Rabiner, 1991] Juang, B. H. and Rabiner, L. R. (1991). Hidden markov models for speech recognition. *Technometrics*, 33(3):251–272. 163
- [Kahou et al., 2016] Kahou, S. E., Bouthillier, X., Lamblin, P., Gulcehre, C., Michalski, V., Konda, K., Jean, S., Froumenty, P., Dauphin, Y., Boulanger-Lewandowski, N., et al. (2016). Emonets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, 10(2):99–111. 162
- [Kanazawa et al., 2018] Kanazawa, A., Black, M. J., Jacobs, D. W., and Malik, J. (2018). End-to-end recovery of human shape and pose. In *Computer Vision and Pattern Recognition (CVPR)*. 170
- [Kanezaki et al., 2018] Kanezaki, A., Matsushita, Y., and Nishida, Y. (2018). Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5010–5019. 83
- [Karlsson et al., 1997] Karlsson, M., Engelbrektsson, P., Hunter, H., O’Niell, A. M., Petrie, H., and Zoldan, D. (1997). Pam-aid. personal adaptive mobility aid for the frail and elderly visually impaired. d3. 1. user requirement study. Technical report, Chalmers University of Technology. 46
- [Kasaei, 2019] Kasaei, H. (2019). Orthographicnet: A deep learning approach for 3d object recognition in open-ended domains. *arXiv preprint arXiv:1902.03057*. 83
- [Kasari and Lawton, 2010] Kasari, C. and Lawton, K. (2010). New directions in behavioral treatment of autism spectrum disorders. *Current Opinion in Neurology*, 23(2):137. 157

- [Kataria and Singh, 2013] Kataria, A. and Singh, M. (2013). A review of data classification using k-nearest neighbour algorithm. *International Journal of Emerging Technology and Advanced Engineering*, 3(6):354–360. 31
- [Kaur et al., 2013] Kaur, M., Gifford, T., Marsh, K. L., and Bhat, A. (2013). Effect of robot–child interactions on bilateral coordination skills of typically developing children and a child with autism spectrum disorder: A preliminary study. *Journal of Motor Learning and Development*, 1(2):31–37. 160
- [Kay et al., 2017] Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al. (2017). The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*. 20, 232
- [Khan et al., 2019] Khan, S. H., Guo, Y., Hayat, M., and Barnes, N. (2019). Unsupervised primitive discovery for improved 3d generative modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9739–9748. 83
- [Kim et al., 2013] Kim, E. S., Berkovits, L. D., Bernier, E. P., Leyzberg, D., Shic, F., Paul, R., and Scassellati, B. (2013). Social robots as embedded reinforcers of social behavior in children with autism. *Journal of autism and developmental disorders*, 43(5):1038–1049. 159
- [Kim et al., 2012] Kim, E. S., Paul, R., Shic, F., and Scassellati, B. (2012). Bridging the research gap: Making hri useful to individuals with autism. *Journal of Human-robot interaction*, 1(1):26–54. 159
- [Koenig and Howard, 2004] Koenig, N. and Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE. 13, 225

- [Komer et al., 2014] Komer, B., Bergstra, J., and Eliasmith, C. (2014). Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn. In *ICML workshop on AutoML*, volume 9. Citeseer. 190
- [Körtgen et al., 2003] Körtgen, M., Park, G.-J., Novotni, M., and Klein, R. (2003). 3d shape matching with 3d shape contexts. In *The 7th central European seminar on computer graphics*, volume 3, pages 5–17. Budmerice. 63
- [Kouroupetroglou et al., 2017] Kouroupetroglou, C., Casey, D., Raciti, M., Barrett, E., D’Onofrio, G., Ricciardi, F., Giuliani, F., Greco, A., Sancarlo, D., Mannion, A., Whelan, S., Pegman, G., Koumpis, A., Riformiati Recupero, D., Kouroupetroglou, A., and Santorelli, A. (2017). Interacting with dementia: The mario approach. In *Harnessing the Power of Technology to Improve Lives*, volume 242 of *Studies in Health Technology and Informatics*, pages 38–47, Netherlands. IOS Press. 46
- [Kozima et al., 2009] Kozima, H., Michalowski, M. P., and Nakagawa, C. (2009). Keepon. *International Journal of Social Robotics*, 1(1):3–18. 159
- [Kozima et al., 2005] Kozima, H., Nakagawa, C., and Yasuda, Y. (2005). Interactive robots for communication-care: A case-study in autism therapy. In *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.*, pages 341–346. IEEE. 159
- [Krafka et al., 2016] Krafka, K., Khosla, A., Kellnhofer, P., Kannan, H., Bhandarkar, S., Matusik, W., and Torralba, A. (2016). Eye tracking for everyone. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 161
- [Krause et al., 2013] Krause, J., Stark, M., Deng, J., and Fei-Fei, L. (2013). 3d object representations for fine-grained categorization. In *Proceedings of the 2013 IEEE International Conference on Computer Vision Workshops, ICCVW ’13*, pages 554–561, Washington, DC, USA. IEEE Computer Society. 187
- [Kriegel et al., 2008] Kriegel, H.-P., Schubert, M., and Zimek, A. (2008). Angle-based outlier detection in high-dimensional data. In *Proceedings*

- of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452. ACM. 199
- [Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc. 33, 56
- [Lai et al., 2011] Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE international conference on robotics and automation*, pages 1817–1824. IEEE. 16, 228
- [Laina et al., 2016] Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. *CoRR*, abs/1606.00373. 137, 141
- [Larriba et al., 2016] Larriba, F., Raya, C., Angulo, C., Albo-Canals, J., Díaz, M., and Boldú, R. (2016). Externalising moods and psychological states in a cloud based system to enhance a pet-robot and child’s interaction. *Biomedical engineering online*, 15(1):72. 159
- [Lathan et al., 2007] Lathan, C., Boser, K., Safos, C., Frentz, C., and Powers, K. (2007). Using cosmo’s learning system (cls) with children with autism. In *Proceedings of the International Conference on Technology-Based Learning with Disabilities*, pages 37–47. 160
- [Lavi et al., 2018] Lavi, B., Serj, M. F., and Ullah, I. (2018). Survey on deep learning techniques for person re-identification task. *CoRR*, abs/1807.05284. 52
- [Le and Duan, 2018] Le, T. and Duan, Y. (2018). Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9204–9214. 36
- [LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444. 32

- [Lee et al., 2009] Lee, H., Grosse, R., Ranganath, R., and Ng, A. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proc. 26th Annual International Conference on Machine Learning*, pages 609–616. ACM. 89
- [Lee et al., 2012] Lee, J., Takehashi, H., Nagai, C., Obinata, G., and Stefanov, D. (2012). Which robot features can stimulate better responses from children with autism in robot-assisted therapy? *International Journal of Advanced Robotic Systems*, 9(3):72. 160
- [Levy-Tzedek et al., 2017] Levy-Tzedek, S., Berman, S., Stiefel, Y., Sharlin, E., Young, J., and Rea, D. (2017). Robotic mirror game for movement rehabilitation. In *2017 International Conference on Virtual Rehabilitation (ICVR)*, pages 1–2. 49
- [Lewis, 1995] Lewis, J. (1995). Fast normalized cross-correlation. In *Vision interface*, volume 10, pages 120–123. 30
- [Li et al., 2018a] Li, J., Chen, B. M., and Hee Lee, G. (2018a). Sonet: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406. 35, 83
- [Li et al., 2018b] Li, M., Shen, F., Wang, J., Guan, C., and Tang, J. (2018b). Person re-identification with activity prediction based on hierarchical spatial-temporal model. *Neurocomputing*, 275:1200 – 1207. 188
- [Li et al., 2018c] Li, M., Zhu, X., and Gong, S. (2018c). Unsupervised person re-identification by deep learning tracklet association. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *Computer Vision – ECCV 2018*, pages 772–788, Cham. Springer International Publishing. 53
- [Li et al., 2012] Li, S., Xu, C., and Xie, M. (2012). A robust o (n) solution to the perspective-n-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1444–1450. 110

- [Li et al., 2014] Li, W., Zhao, R., Xiao, T., and Wang, X. (2014). Deep-reid: Deep filter pairing neural network for person re-identification. In *CVPR*, pages 152–159. IEEE Computer Society. 53
- [Liang et al., 2002] Liang, K. M., Rajeswari, M., and Khoo, B. E. (2002). Similarity measure determination from nurbs-warping method. In *7th International Conference on Control, Automation, Robotics and Vision, 2002. ICARCV 2002.*, volume 3, pages 1222–1227. IEEE. 58
- [Liang et al., 2003] Liang, K. M., Rajeswari, M., and Khoo, B. E. (2003). Nurbs: A new shape descriptor for shape-based image retrieval. Technical report, Technical report, University Science Malaysia. 58
- [Lin et al., 2014a] Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014a). Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312. 192
- [Lin et al., 2014b] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014b). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer. 172
- [Liu et al., 2015] Liu, A., Wang, Z., Nie, W., and Su, Y. (2015). Graph-based characteristic view set extraction and matching for 3d model retrieval. *Information Sciences*, 320:429–442. 16, 228
- [Liu et al., 2008] Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE. 199
- [Liu et al., 2018a] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., and Pietikäinen, M. (2018a). Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*. 56
- [Liu et al., 2018b] Liu, S., Giles, L., and Ororbia, A. (2018b). Learning a hierarchical latent-variable model of 3d shapes. In *2018 International Conference on 3D Vision (3DV)*, pages 542–551. IEEE. 83

- [Liu et al., 2019a] Liu, X., Han, Z., Liu, Y.-S., and Zwicker, M. (2019a). Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8778–8785. 83
- [Liu et al., 2019b] Liu, Z., Tang, H., Lin, Y., and Han, S. (2019b). Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, pages 963–973. 35
- [Louie et al., 2014] Louie, W. G., Vaquero, T., Nejat, G., and Beck, J. C. (2014). An autonomous assistive robot for planning, scheduling and facilitating multi-user activities. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5292–5298. 47
- [Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee. 30, 43
- [Luo et al., 2020] Luo, F., Zhang, L., Du, B., and Zhang, L. (2020). Dimensionality reduction with enhanced hybrid-graph discriminant learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*. 33
- [Luxton and Riek, 2019] Luxton, D. D. and Riek, L. D. (2019). Artificial intelligence and robotics in rehabilitation. In *Handbook of rehabilitation psychology (3rd ed.)*, pages 507–520. American Psychological Association. 151
- [Lytridis et al., 2018] Lytridis, C., Vrochidou, E., Chatzistamatis, S., and Kaburlasos, V. (2018). Social engagement interaction games between children with autism and humanoid robot nao. In *The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications*, pages 562–570. Springer. 159

- [Ma et al., 2017] Ma, C., An, W., Lei, Y., and Guo, Y. (2017). Bv-cnns: Binary volumetric convolutional networks for 3d object recognition. In *BMVC*, volume 1, page 4. 83
- [Mandelbrot, 1993] Mandelbrot, B. (1993). How long is the coast of britain? statistical self-similarity and fractional dimension. *Classics on fractals, edited by GE Edgar*, pages 351–358. 71
- [Martin-Rico et al., 2019] Martin-Rico, F., Gomez-Donoso, F., Escalona, F., Cazorla, M., and Garcia-Rodriguez, J. (2019). Artificial semantic memory with autonomous learning applied to social robots. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 401–411. Springer. 207, 243
- [Martin-Rico et al., 2020] Martin-Rico, F., Gomez-Donoso, F., Escalona, F., Garcia-Rodriguez, J., and Cazorla, M. (2020). Semantic visual recognition in a cognitive architecture for social robots. *Integrated Computer-Aided Engineering*, (Preprint):1–16. 206, 242
- [Martínez-Gómez et al., 2016] Martínez-Gómez, J., Morell, V., Cazorla, M., and García-Varea, I. (2016). Semantic localization in the pcl lib. *Robot. and Autonomous Systems*, 75, Part B:641 – 648. 92
- [Martinez-Martin and Cazorla, 2019] Martinez-Martin, E. and Cazorla, M. (2019). Rehabilitation technology: Assistance from hospital to home. *Computational Intelligence and Neuroscience*, 2019:1–8. 151
- [Martinez-Martin et al., 2019] Martinez-Martin, E., Costa, A., and Cazorla, M. (2019). PHAROS 2.0—a PHysical assistant RObot system improved. *Sensors*, 19(20):4531. 153
- [Martinez-Martin and del Pobil, 2017a] Martinez-Martin, E. and del Pobil, A. P. (2017a). Personal robot assistants for elderly care: An overview. In Costa, A., Julian, V., and Novais, P., editors, *Personal Assistants: Emerging Computational Technologies*, pages 77–91. Springer. 49

- [Martinez-Martin and del Pobil, 2017b] Martinez-Martin, E. and del Pobil, A. P. (2017b). Personal robot assistants for elderly care: An overview. In *Intelligent Systems Reference Library*, pages 77–91. Springer International Publishing. 153
- [Martinez-Martin et al., 2020] Martinez-Martin, E., Escalona, F., and Cazorla, M. (2020). Socially assistive robots for older adults and people with autism: An overview. *Electronics*, 9(2):367. 206, 242
- [Masi et al., 2018] Masi, I., Wu, Y., Hassner, T., and Natarajan, P. (2018). Deep face recognition: A survey. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 471–478. IEEE. 162
- [Matas et al., 2004] Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767. 30
- [Maturana and Scherer, 2015] Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE. 36
- [Mavadati et al., 2016] Mavadati, S. M., Feng, H., Salvador, M., Silver, S., Gutierrez, A., and Mahoor, M. H. (2016). Robot-based therapeutic protocol for training children with autism. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 855–860. IEEE. 159, 164
- [McGlaun, 2011] McGlaun, S. (2011). *Asus gets Xtion Pro Live ready for launch.* <https://www.slashgear.com/asus-gets-xtion-pro-live-ready-for-launch-19165977/> (Accessed: 2017-04-29). 4, 216
- [Meldrum et al., 2012] Meldrum, D., Glennon, A., Herdman, S., Murray, D., and McConn-Walsh, R. (2012). Virtual reality rehabilitation of balance: assessment of the usability of the nintendo wii® fit plus. *Disability and Rehabilitation: Assistive Technology*, 7(3):205–210. 50

- [Moerwald, 2019] Moerwald, T. (2019). Point cloud library - trimble code sprint final report. <https://modelnet.cs.princeton.edu/>. Accessed: 2019-12-01. 58
- [Mollahosseini et al., 2018] Mollahosseini, A., Abdollahi, H., and Mahoor, M. H. (2018). Studying effects of incorporating automated affect perception with spoken dialog in social robots. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 783–789. IEEE. 160
- [Monfort et al., 2019] Monfort, M., Andonian, A., Zhou, B., Ramakrishnan, K., Bargal, S. A., Yan, T., Brown, L., Fan, Q., Gutfreund, D., Vondrick, C., et al. (2019). Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):502–508. 20, 232
- [Monge and Postolache, 2018] Monge, J. and Postolache, O. (2018). Augmented reality and smart sensors for physical rehabilitation. In *10th International Conference and Exposition on Electrical and Power Engineering (EPE2018)*, pages 1010–1014. 50
- [Moyle et al., 2016] Moyle, W., Jones, C., Sung, B., Bramble, M., O’Dwyer, S., Blumenstein, M., and Estivill-Castro, V. (2016). What effect does an animal robot called cuddler have on the engagement and emotional response of older people with dementia? A pilot feasibility study. *I. J. Social Robotics*, 8(1):145–156. 46
- [Muja and Lowe, 2014] Muja, M. and Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240. 44
- [Mur-Artal and Tardos, 2017] Mur-Artal, R. and Tardos, J. D. (2017). ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. *IEEE Transactions on Robotics*, 33:1255–1262. 169

- [Mura and Carta, 2013] Mura, G. and Carta, M. G. (2013). Physical activity in depressed elderly. a systematic review. *Clinical Practice & Epidemiology in Mental Health*, 9(1):125–135. 153
- [Naseer et al., 2019] Naseer, M., Khan, S., and Porikli, F. (2019). Indoor scene understanding in 2.5/3d for autonomous agents: A survey. *IEEE Access*, 7:1859–1887. 55
- [Nintendo, 2008] Nintendo (2008). Wii fit. <https://www.nintendo.es/Juegos/Wii/Wii-Fit-283894.html>. 50
- [Nunez et al., 2015] Nunez, E., Matsuda, S., Hirokawa, M., and Suzuki, K. (2015). Humanoid robot assisted training for facial expressions recognition based on affective feedback. In *International Conference on Social Robotics*, pages 492–501. Springer. 159, 163
- [Oh et al., 2019] Oh, S., Oh, Y. H., and Ju, D. Y. (2019). Understanding the preference of the elderly for companion robot design. In *Advances in Intelligent Systems and Computing*, pages 92–103. Springer International Publishing. 153
- [Olver, 1995] Olver, P. J. (1995). *Equivalence, invariants and symmetry*. Cambridge University Press. 30
- [Osada et al., 2001] Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. (2001). Matching 3d models with shape distributions. In *Proceedings International Conference on Shape Modeling and Applications*, pages 154–166. IEEE. 34
- [Pagani and Scott, 2018] Pagani, L. and Scott, P. J. (2018). Curvature based sampling of curves and surfaces. *Computer Aided Geometric Design*, 59:32–48. 59
- [Palmqvist and Danielsson, 2019] Palmqvist, L. and Danielsson, H. (2019). Parents act as intermediary users for their children when using assistive technology for cognition in everyday planning: Results from a parental survey. *Assistive Technology*, pages 1–9. 150

- [Pant et al., 2018] Pant, P., Gupta, V., Khanna, A., and Saxena, N. (2018). Technology foresight study on assistive technology for locomotor disability. *Technology and Disability*, 29(4):163–171. 150
- [Papazov and Burschka, 2011] Papazov, C. and Burschka, D. (2011). An efficient ransac for 3d object recognition in noisy and occluded scenes. *Computer Vision–ACCV 2010*, pages 135–148. 45
- [Perera et al., 2017] Perera, V., Pereira, T., Connell, J., and Veloso, M. M. (2017). Setting up pepper for autonomous navigation and personalized interaction with users. *CoRR*, abs/1704.04797. 126
- [Petric et al., 2017] Petric, F., Miklic, D., and Kovacic, Z. (2017). Robot-assisted autism spectrum disorder diagnostics using pomdps. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 369–370. ACM. 159
- [Phillips and Proulx, 2018] Phillips, M. and Proulx, M. J. (2018). Social interaction without vision: An assessment of assistive technology for the visually impaired. *Technology & Innovation*, 20(1):85–93. 150
- [Piegl, 1991] Piegl, L. (1991). On nurbs: a survey. *IEEE Computer Graphics and Applications*, 11(1):55–71. 57, 58
- [Piegl and Tiller, 1997] Piegl, L. and Tiller, W. (1997). *The NURBS book*. Springer Science & Business Media. 58
- [Piegl and Tiller, 1999] Piegl, L. A. and Tiller, W. (1999). Computing offsets of nurbs curves and surfaces. *Computer-Aided Design*, 31(2):147–156. 59
- [Pineau et al., 2003] Pineau, J., Montemerlo, M., Pollack, M., Roy, N., and Thrun, S. (2003). Towards robotic assistants in nursing homes: Challenges and results. *Robotics and Autonomous Systems*, 42(3):271 – 281. Socially Interactive Robots. 46
- [PointClouds, 2014] PointClouds (2014). *The PCL Registration API*. http://pointclouds.org/documentation/tutorials/registration_api.php (Accessed: 2017-04-30). 43

- [PointClouds, 2015] PointClouds (2015). *PCLVisualizer*. http://pointclouds.org/documentation/tutorials/pcl_visualizer.php (Accessed: 2017-04-30). 16, 228
- [Pollack et al., 2003] Pollack, M. E., Brown, L., Colbry, D., McCarthy, C. E., Orosz, C., Peintner, B., Ramakrishnan, S., and Tsamardinos, I. (2003). Autominder: an intelligent cognitive orthotic system for people with memory impairment. *Robotics and Autonomous Systems*, 44(3):273 – 282. Best papers presented at IAS-7. 46
- [Pope, 1994] Pope, A. R. (1994). Model-based object recognition. *A Survey of Recent Techniques, Technical Report*. 28
- [Pradel et al., 2010] Pradel, G., Dansart, P., Puret, A., and Barthélemy, C. (2010). Generating interactions in autistic spectrum disorders by means of a mobile robot. In *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*, pages 1540–1545. IEEE. 160
- [Pramerdorfer and Kampel, 2016] Pramerdorfer, C. and Kampel, M. (2016). Facial expression recognition using convolutional neural networks: state of the art. *arXiv preprint arXiv:1612.02903*. 162
- [Pronobis et al., 2009] Pronobis, A., Martinez Mozos, O., Caputo, B., and Jensfelt, P. (2009). Multi-modal semantic place classification. *The International Journal of Robotics Research*. 38
- [PTC, 2019] PTC, I. (2019). Vuforia. <https://developer.vuforia.com/>. 186
- [Qi et al., 2019] Qi, C. R., Litany, O., He, K., and Guibas, L. J. (2019). Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9277–9286. 35
- [Qi et al., 2016] Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., and Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656. 37

- [Qi et al., 2017] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108. 35
- [Qian et al., 2019] Qian, X., Fu, Y., Xiang, T., Jiang, Y., and Xue, X. (2019). Leader-based multi-scale attention deep architecture for person re-identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1. 52
- [Qidwai et al., 2019] Qidwai, U., Kashem, S. B. A., and Conor, O. (2019). Humanoid robot as a teacher’s assistant: Helping children with autism to learn social and academic skills. *Journal of Intelligent & Robotic Systems*, pages 1–12. 159
- [Quigley et al., 2009a] Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009a). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*. 113
- [Quigley et al., 2009b] Quigley, M., Faust, J., Foote, T., and Leibs, J. (2009b). Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. 12, 224
- [Rafiei and Adeli, 2017] Rafiei, M. H. and Adeli, H. (2017). A new neural dynamic classification algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 28(12):3074–3083. 202
- [Ravindra et al., 2009] Ravindra, P., De Silva, S., Tadano, K., Saito, A., Lambacher, S. G., and Higashi, M. (2009). Therapeutic-assisted robot for children with autism. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3561–3567. IEEE. 160
- [Redmon, 2016] Redmon, J. (2013–2016). Darknet: Open source neural networks in c. <http://pjreddie.com/darknet/>. 15
- [Redmon and Farhadi, 2018] Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *CoRR*, abs/1804.02767. 115

- [Redmon and Farhadi, 2018] Redmon, J. and Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *ArXiv e-prints*. 171, 173
- [Redmon and Farhadi, 2018] Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv*. 189
- [Rentschler et al., 2003] Rentschler, A., Cooper, R., Blasch, B., and Boninger, M. (2003). Intelligent walkers for the elderly: Performance and safety testing of va-pamaid robotic walker. *Journal of Rehabilitation Research and Development*, 40(5):423 – 431. 46
- [Rian and Sassone, 2014] Rian, I. M. and Sassone, M. (2014). Fractal-based generative design of structural trusses using iterated function system. *International Journal of Space Structures*, 29(4):181–203. 72
- [Ricks and Colton, 2010] Ricks, D. J. and Colton, M. B. (2010). Trends and considerations in robot-assisted autism therapy. In *2010 IEEE international conference on robotics and automation*, pages 4354–4359. IEEE. 158
- [Riisgaard and Blas, 2005] Riisgaard, S. and Blas, M. R. (2005). Slam for dummies: A tutorial approach to simultaneous localization and mapping. Technical report, MIT. 39, 40, 41, 42
- [Robins et al., 2009] Robins, B., Dautenhahn, K., and Dickerson, P. (2009). From isolation to communication: a case study evaluation of robot assisted play for children with autism with a minimally expressive humanoid robot. In *2009 Second International Conferences on Advances in Computer-Human Interactions*, pages 205–211. IEEE. 159
- [Robins et al., 2006] Robins, B., Dautenhahn, K., and Dubowski, J. (2006). Does appearance matter in the interaction of children with autism with a humanoid robot? *Interaction studies*, 7(3):479–512. 158
- [Robins et al., 2004a] Robins, B., Dautenhahn, K., Te Boekhorst, R., and Billard, A. (2004a). Effects of repeated exposure to a humanoid robot on children with autism. In *Designing a more inclusive world*, pages 225–236. Springer. 160, 161

- [Robins et al., 2004b] Robins, B., Dickerson, P., Stribling, P., and Dautenhahn, K. (2004b). Robot-mediated joint attention in children with autism: A case study in robot-human interaction. *Interaction studies*, 5(2):161–198. 160, 161
- [Robocup, 2019] Robocup (2019 (accessed October 24, 2019)). *RoboCup@Home 2019*. 191
- [Rodriguez et al., 2016] Rodriguez, A., Gomez-Donoso, F., Martinez-Gomez, J., and Cazorla, M. (2016). Building 3d maps with tag information. In *XVII Workshop en Agentes Físicos*. 102
- [ROS, 2014] ROS (2014). *Pepper Tutorials*. <http://wiki.ros.org/pepper/Tutorials>. 11, 223
- [ROS, 2018] ROS (2018). *move_base*. http://wiki.ros.org/move_base (Accessed: 2018-09-02). 126, 127
- [Roth and Winter, 2008] Roth, P. M. and Winter, M. (2008). Survey of appearance-based methods for object recognition. *Inst. for Computer Graphics and Vision, Graz University of Technology, Austria, Technical Report ICGTR0108 (ICG-TR-01/08)*. 30
- [Rousseeuw and Driessen, 1999] Rousseeuw, P. J. and Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223. 199
- [Ruano et al., 2019] Ruano, A., Hernandez, A., Ureña, J., Ruano, M., and Garcia, J. (2019). NILM techniques for intelligent home energy management and ambient assisted living: A review. *Energies*, 12(11):2203. 151
- [Russakovsky et al., 2015] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252. 56
- [Rusu et al., 2009] Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (fpfh) for 3d registration. In *ICRA '09*. 34, 44

- [Rusu et al., 2010] Rusu, R. B., Bradski, G., Thibaux, R., and Hsu, J. (2010). Fast 3d recognition and pose using the viewpoint feature histogram. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162. IEEE. 34
- [Rusu and Cousins, 2011] Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1–4. IEEE. 16, 228
- [Salvador et al., 2016] Salvador, M., Marsh, A. S., Gutierrez, A., and Mahoor, M. H. (2016). Development of an aba autism intervention delivered by a humanoid robot. In *International Conference on Social Robotics*, pages 551–560. Springer. 159
- [Salvador et al., 2015] Salvador, M. J., Silver, S., and Mahoor, M. H. (2015). An emotion recognition comparative study of autistic and typically-developing children using the zeno robot. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6128–6133. IEEE. 159
- [Salvador and Chan, 2007] Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580. 176
- [Sandler et al., 2018] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. 192
- [Scassellati, 2007] Scassellati, B. (2007). How social robots will help us to diagnose, treat, and understand autism. In *Robotics research*, pages 552–563. Springer. 157
- [Scassellati et al., 2012] Scassellati, B., Admoni, H., and Matarić, M. (2012). Robots for use in autism research. *Annual review of biomedical engineering*, 14:275–294. 157
- [Schroeder et al., 2005] Schroeder, W., Martin, K., and Lorensen, B. (2005). An object-oriented approach to 3-d graphics. 16, 228

- [Scovanner et al., 2007] Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proc. 15th ACM int. conf. on Multimedia*. 43
- [Sewell and Komogortsev, 2010] Sewell, W. and Komogortsev, O. (2010). Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 3739–3744. ACM. 161
- [Shamsuddin et al., 2012] Shamsuddin, S., Yussof, H., Ismail, L., Hanapiah, F. A., Mohamed, S., Piah, H. A., and Zahari, N. I. (2012). Initial response of autistic children in human-robot interaction therapy with humanoid robot nao. In *2012 IEEE 8th International Colloquium on Signal Processing and its Applications*, pages 188–193. IEEE. 159
- [Sharma et al., 2016] Sharma, A., Grau, O., and Fritz, M. (2016). Vconvdae: Deep volumetric shape learning without object labels. In *European Conference on Computer Vision*, pages 236–250. Springer. 37
- [Shen, 2002] Shen, G. (2002). Fractal dimension and fractal growth of urbanized areas. *International Journal of Geographical Information Science*, 16(5):419–437. 37
- [Shen et al., 2018] Shen, Y., Li, H., Yi, S., Chen, D., and Wang, X. (2018). Person re-identification with deep similarity-guided graph neural network. In *ECCV (15)*, volume 11219 of *Lecture Notes in Computer Science*, pages 508–526. Springer. 53
- [Silva et al., 2017] Silva, V., Leite, P., Soares, F., Esteves, J. S., and Costa, S. (2017). Imitate me!—preliminary tests on an upper members gestures recognition system. In *CONTROLO 2016*, pages 373–383. Springer. 159
- [SilverFit, 2019] SilverFit (2019). Silverfit 3d. <https://silverfit.com/>. 51
- [Simon et al., 2017] Simon, T., Joo, H., Matthews, I., and Sheikh, Y. (2017). Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*. 119

- [Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556. 192
- [Sims et al., 2019] Sims, S. D., Putnam, V., and Conati, C. (2019). Predicting confusion from eye-tracking data with recurrent neural networks. *arXiv preprint arXiv:1906.11211*. 161
- [Singh et al., 2014] Singh, A., Sha, J., Narayan, K. S., Achim, T., and Abbeel, P. (2014). Bigbird: A large-scale 3d database of object instances. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 509–516. IEEE. 16, 228
- [Sivic and Zisserman, 2003] Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *Ninth IEEE Int. Conf. on Computer Vision*. 92
- [Smith, 2002] Smith, L. I. (2002). A tutorial on principal components analysis. Technical report. 60
- [Song et al., 2015] Song, S., Lichtenberg, S. P., and Xiao, J. (2015). Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, pages 567–576. IEEE Computer Society. 17, 229
- [Sousa et al., 2016] Sousa, M., Vieira, J., Medeiros, D., Arsenio, A., and Jorge, J. (2016). Sleevear: Augmented reality for rehabilitation using realtime feedback. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pages 175–185, Sonoma, California, USA. 51
- [Srinivasa et al., 2010] Srinivasa, S., Ferguson, D., Helfrich, C., Berenson, D., Romea, A. C., Diankov, R., Gallagher, G., Hollinger, G., Kuffner, J., and Vandeweghe, J. M. (2010). Herb: a home exploring robotic butler. *Autonomous Robots*, 28(1). 47
- [Srinivasa et al., 2012] Srinivasa, S. S., Berenson, D., Cakmak, M., Collet, A., Dogar, M. R., Dragan, A. D., Knepper, R. A., Niemueller, T., Strabala, K., Weghe, M. V., and Ziegler, J. (2012). Herb 2.0: Lessons

- learned from developing a mobile manipulator for the home. *Proceedings of the IEEE*, 100(8):2410–2428. 47
- [Srinivasan et al., 2013] Srinivasan, S. M., Lynch, K. A., Bubela, D. J., Gifford, T. D., and Bhat, A. N. (2013). Effect of interactions between a child and a robot on the imitation and praxis performance of typically developing children and a child with autism: A preliminary study. *Perceptual and Motor Skills*, 116(3):885–904. 160
- [Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958. 79
- [Stefano et al., 2014] Stefano, M., Patrizia, P., Mario, A., Ferlini, G., Rizzello, R., and Rosati, G. (2014). Robotic upper limb rehabilitation after acute stroke by nerebot: Evaluation of treatment costs. *BioMed Research International*, 2014. 49
- [Steffen et al., 2013] Steffen, D., Bleser, G., Weber, M., Stricker, D., Fradet, L., and Marin, F. (2013). A personalized exercise trainer for the elderly. *Journal of Ambient Intelligence and Smart Environments*, 5:547–562. 49
- [Su et al., 2018] Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.-H., and Kautz, J. (2018). Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539. 35
- [Su et al., 2015] Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953. 36
- [Sun et al., 2015] Sun, Y., Liang, D., Wang, X., and Tang, X. (2015). Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*. 162

- [Suresh et al., 2018] Suresh, A., Arora, C., Laha, D., Gaba, D., and Bhambri, S. (2018). Intelligent smart glass for visually impaired using deep learning machine vision techniques and robot operating system (ROS). In *Robot Intelligence Technology and Applications 5*, pages 99–112. Springer International Publishing. 150
- [Suykens and Vandewalle, 1999] Suykens, J. A. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300. 31
- [Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proc. IEEE conference on CVPR*. 88, 92
- [Tahsin et al., 2016] Tahsin, M. M., Khan, R., and Gupta, A. K. S. (2016). Assistive technology for physically challenged or paralyzed person using voluntary tongue movement. In *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*. IEEE. 150
- [Tapus and Mataric, 2008] Tapus, A. and Mataric, M. J. (2008). Socially assistive robots: The link between personality, empathy, physiological signals, and task performance. In *AAAI spring symposium: emotion, personality, and social behavior*, pages 133–140. 152
- [Tapus et al., 2012] Tapus, A., Peca, A., Aly, A., Pop, C., Jisa, L., Pintea, S., Rusu, A. S., and David, D. O. (2012). Children with autism social engagement in interaction with nao, an imitative robot: A series of single case experiments. *Interaction studies*, 13(3):315–347. 159
- [Tatarchenko et al., 2017] Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2017). Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096. 37
- [Thrun et al., 2002] Thrun, S. et al. (2002). Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, pages 1–35. 38

- [Tombari et al., 2010] Tombari, F., Salti, S., and Di Stefano, L. (2010). Unique signatures of histograms for local surface description. In *Computer Vision–ECCV 2010*, pages 356–369. Springer. 63
- [Torres-Camara et al., 2019] Torres-Camara, J. M., Escalona, F., Gomez-Donoso, F., and Cazorla, M. (2019). Map slammer: Densifying scattered kslam 3d maps with estimated depth. In *Iberian Robotics conference*, pages 563–574. Springer. 208, 244
- [Turk and Pentland, 1991] Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86. 162
- [Tyromotion, 2018] Tyromotion (2018). Amadeo. <https://tyromotion.com/en/produkte/amadeo/>. 49
- [Tzafestas, 2016] Tzafestas, S. (2016). Sociorobot field studies. In *Sociorobot World*, pages 175–202. Springer. 160
- [Valenti et al., 2011] Valenti, R., Sebe, N., and Gevers, T. (2011). Combining head pose and eye location information for gaze estimation. *IEEE Transactions on Image Processing*, 21(2):802–815. 161
- [Van der Loos et al., 1999] Van der Loos, H. M., Wagner, J. J., Smaby, N., Chang, K., Madrigal, O., Leifer, L. J., and Khatib, O. (1999). Provar assistive robot system architecture. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 1, pages 741–746. IEEE. 46
- [Viola et al., 2001] Viola, P., Jones, M., et al. (2001). Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, 1(511-518):3. 162
- [Wainer et al., 2014a] Wainer, J., Dautenhahn, K., Robins, B., and Amirabdollahian, F. (2014a). A pilot study with a novel setup for collaborative play of the humanoid robot kaspar with children with autism. *International journal of social robotics*, 6(1):45–65. 159

- [Wainer et al., 2014b] Wainer, J., Robins, B., Amirabdollahian, F., and Dautenhahn, K. (2014b). Using the humanoid robot kaspar to autonomously play triadic games and facilitate collaborative play among children with autism. *IEEE Transactions on Autonomous Mental Development*, 6(3):183–199. 159
- [Wang et al., 2018a] Wang, C., Zhang, Q., Huang, C., Liu, W., and Wang, X. (2018a). Mancs: A multi-task attentional network with curriculum sampling for person re-identification. In *ECCV*. 52
- [Wang et al., 2018b] Wang, K., Wang, H., Liu, M., and Xing, X. (2018b). Survey on person re-identification based on deep learning. *CAAI Transactions on Intelligence Technology*. 53
- [Wang et al., 2017] Wang, P.-S., Liu, Y., Guo, Y.-X., Sun, C.-Y., and Tong, X. (2017). O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):1–11. 36
- [Wang et al., 2014] Wang, T., Gong, S., Zhu, X., and Wang, S. (2014). Person re-identification by video ranking. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, pages 688–703, Cham. Springer International Publishing. 53
- [Wang et al., 2011] Wang, T., Zhang, H., Ma, X., Zhu, Y., Zhou, Z., and Qian, B. (2011). A home nursing robot system. In Zeng, D., editor, *Future Intelligent Information Systems*, pages 317–324, Berlin, Heidelberg. Springer Berlin Heidelberg. 47
- [Wang et al., 2006] Wang, W., Pottmann, H., and Liu, Y. (2006). Fitting b-spline curves to point clouds by curvature-based squared distance minimization. *ACM Transactions on Graphics (ToG)*, 25(2):214–238. 58
- [Watanabe and Yoneda, 2009] Watanabe, K. and Yoneda, Y. (2009). The world’s smallest biped humanoid robot “i-sobot”. In *2009 IEEE Workshop on Advanced Robotics and its Social Impacts*, pages 51–53. IEEE. 160

- [Wei et al., 2016] Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional pose machines. In *CVPR*. 119
- [Werbos and Werbos, 1994] Werbos, P. and Werbos, P. (1994). Beyond regression: New tools for prediction and analysis in the behavioral sciences. the roots of backpropagation. 32
- [Wilson et al., 2019] Wilson, G., Pereyda, C., Raghunath, N., de la Cruz, G., Goel, S., Nesaei, S., Minor, B., Schmitter-Edgecombe, M., Taylor, M., and Cook, D. J. (2019). Robot-enabled support of daily activities in smart home environments. In *Cognitive Systems Research*, volume 54, pages 258–272. Elsevier. 155
- [Wohlkinger and Vincze, 2011] Wohlkinger, W. and Vincze, M. (2011). Ensemble of shape functions for 3d object classification. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, pages 2987–2992. IEEE. 34
- [Wold et al., 1987] Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52. 199
- [Wood et al., 2019] Wood, L. J., Zarakı, A., Robins, B., and Dautenhahn, K. (2019). Developing kaspar: a humanoid robot for children with autism. *International Journal of Social Robotics*, pages 1–18. 159
- [Wood et al., 2017] Wood, L. J., Zarakı, A., Walters, M. L., Novanda, O., Robins, B., and Dautenhahn, K. (2017). The iterative development of the humanoid robot kaspar: An assistive robot for children with autism. In *International Conference on Social Robotics*, pages 53–63. Springer. 159
- [Woodyard et al., 2015] Woodyard, A. H., Guleksen, E. P., and Lindsay, R. O. (2015). Pabi: Developing a new robotic platform for autism therapy. Technical report. 159
- [World Health Organization, 2010] World Health Organization (2010). Community-based rehabilitation: cbr guidelines. Technical report, Geneva. 48

- [World Health Organization, 2015] World Health Organization (2015). Who global disability action plan 2014-2021. better health for all people with disability. Technical report. 167
- [World Health Organization (WHO), 2017] World Health Organization (WHO) (2017). Global strategy and action plan on ageing and health. Technical report. 152
- [Wu et al., 2016] Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pages 82–90. 37
- [Wu et al., 2015a] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015a). 3d shapenets: A deep representation for volumetric shapes. pages 1912–1920. 17, 36, 229
- [Wu et al., 2015b] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015b). 3d shapenets: A deep representation for volumetric shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 17, 64, 83, 229
- [Wu et al., 2019] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2019). Modelnet benchmark leaderboard. <https://modelnet.cs.princeton.edu/>. Accessed: 2019-12-01. 36, 66, 67, 81
- [Xia et al., 2020] Xia, S., Chen, D., Wang, R., Li, J., and Zhang, X. (2020). Geometric primitives in lidar point clouds: A review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:685–707. 33
- [Xiang et al., 2016] Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., and Savarese, S. (2016). Objectnet3d: A large scale database for 3d object recognition. In *European Conference Computer Vision (ECCV)*. 17, 229
- [Xiong et al., 2018] Xiong, W., Wu, L., Allewa, F., Droppo, J., Huang, X., and Stolcke, A. (2018). The microsoft 2017 conversational speech

- recognition system. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5934–5938. IEEE. 163
- [Xu and Todorovic, 2016] Xu, X. and Todorovic, S. (2016). Beam search for learning a deep convolutional neural network of 3d shapes. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3506–3511. IEEE. 62
- [Yabuki and Sumi, 2018] Yabuki, K. and Sumi, K. (2018). Learning support system for effectively conversing with individuals with autism using a humanoid robot. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 4266–4270. IEEE. 159, 164
- [Yang et al., 2016] Yang, S., Maturana, D., and Scherer, S. (2016). Real-time 3d scene layout from a single image using convolutional neural networks. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2183–2189. 169
- [Yang et al., 2018] Yang, X., Shyu, M.-L., Yu, H.-Q., Sun, S.-M., Yin, N.-S., and Chen, W. (2018). Integrating image and textual information in human–robot interactions for children with autism spectrum disorder. *IEEE Transactions on Multimedia*, 21(3):746–759. 159, 164
- [Yavartanoo et al., 2018] Yavartanoo, M., Kim, E. Y., and Lee, K. M. (2018). Spnet: Deep 3d object classification and retrieval using stereographic projection. In *Asian Conference on Computer Vision*, pages 691–706. Springer. 83
- [Yoshikawa et al., 2019] Yoshikawa, Y., Kumazaki, H., Matsumoto, Y., Miyao, M., Kikuchi, M., and Ishiguro, H. (2019). Relaxing gaze aversion of adolescents with autism spectrum disorder in consecutive conversations with human and android robot—a preliminary study—. *Frontiers in psychiatry*, 10:370. 161
- [Yu et al., 2018] Yu, R., Dou, Z., Bai, S., Zhang, Z., Xu, Y., and Bai, X. (2018). Hard-aware point-to-set deep metric for person re-identification. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors,

- Computer Vision – ECCV 2018*, pages 196–212, Cham. Springer International Publishing. 52
- [Yucel et al., 2018] Yucel, M. K., Bilge, Y. C., Oguz, O., Ikizler-Cinbis, N., Duygulu, P., and Cinbis, R. G. (2018). Wildest faces: Face detection and recognition in violent settings. *arXiv preprint arXiv:1805.07566*. 162
- [ZeroC, 2018] ZeroC (2018). Internet Communications Engine. <https://github.com/zeroc-ice/ice>. Online; accessed 17 October 2018. 113
- [Zheng et al., 2016] Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C., Wang, S., and Tian, Q. (2016). Mars: A video benchmark for large-scale person re-identification. 53
- [Zheng et al., 2015] Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., and Tian, Q. (2015). Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124. 53
- [Zheng et al., 2017] Zheng, Z., Zheng, L., and Yang, Y. (2017). Unlabeled samples generated by GAN improve the person re-identification baseline in vitro. *CoRR*, abs/1701.07717. 53
- [Zhou et al., 2018a] Zhou, B., Wu, K., Lv, P., Wang, J., Chen, G., Ji, B., and Liu, S. (2018a). A new remote health-care system based on moving robot intended for the elderly at home. *Journal of healthcare engineering*, 2018. 48
- [Zhou et al., 2018b] Zhou, Q.-Y., Park, J., and Koltun, V. (2018b). Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*. 76
- [Zitova and Flusser, 2003] Zitova, B. and Flusser, J. (2003). Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000. 44

Glossary

AAL Ambient Assisted Living. 87, 105–107, 110–112, 116, 118, 119, 151

AI Artificial Intelligence. 152, 164, 165

AMCL Adaptive Monte Carlo Localization. 126

AOI Area Of Interest. 167, 172, 173, 181, 182, 189, 192, 201

AR Assistive Robotics. 151, 152, 165

BAIR Berkeley Artificial Intelligence Research Laboratory. 14, 226

BSD Berkeley Software Distribution. 12, 14, 15, 224, 226, 227

CAD 3D Computer Aided Design. 56

Caffe Convolution Architecture For Feature Extraction. 14, 226

CNN Convolutional Neural Network. 32, 53, 98, 121, 153, 161–163, 172, 187, 189

CPM Convolutional Pose Machines. 119

CPU Central Processing Unit. 15

CUDA Compute Unified Device Architecture. 15, 191, 227

DL Deep Learning. 15, 52, 53, 149, 187–191, 199, 200, 202, 227

DT Decision Tree. 193, 194

- DTW** Dynamic Time Warping. 173, 174, 181, 184
- EKF** Extended Kalman Filter. 39
- FLANN** Fast Library Approximate Nearest Neighbor. 44
- FPFH** Fast Point Feature Histogram. 44
- GPU** Graphics Processing Unit. 14, 15, 226
- GS** Grid Search. 190, 192, 193
- HMR** Human Mesh Recovery. 170, 173, 181, 183
- HRI** Human-Robot Interaction. 151, 153
- ICE** Internet Communications Engine. 113
- JCR** Journal Citation Reports. 205, 242
- KARD** Kinect Action Recognition Dataset. 20, 181, 182, 184, 194, 196, 233
- KNN** K-Nearest Neighbors. 76, 77, 121, 122, 192–200
- LBPH** Local Binary Pattern Histograms. 162
- LiDAR** Laser Imaging Detection and Ranging. 56
- ML** Machine Learning. 15, 151, 152, 158, 160, 227
- MPS** Motion Planning System. 122, 125–127, 131
- NB** Naive Bayes. 193–200
- NHS** British National Health Security. 180
- OGT** Obstacles over the Ground Tracker. 109, 110, 112–114, 116–118

- PCA** Principal Component Analysis. 162
- PCD** Point Cloud Data. 17, 18, 229, 230
- PCL** Point Cloud Library. 15–17, 227–229
- PFH** Point Feature Histogram. 44
- PIMS** Person Identification Memory System. 188, 190, 202
- RANSAC** Random Sample Consensus. 45, 108, 113
- RCNN** Region-based Convolutional Neural Network. 109, 189
- RF** Random Forest. 192–200
- RMSE** Root Mean Square Error. 136, 138, 144, 145
- RNN** Recurrent Neural Network. 153, 161, 163
- ROS** Robot Operating System. 11–14, 16, 90, 98, 111, 113, 125–127, 132, 223–225, 228
- SAR** Socially Assistive Robotics. 151–153, 157, 158, 162, 164, 165
- SGGNN** Similarity-Guided Graph Neural Network. 53
- SIFT** Scale Invariant Feature Transform. 43
- SIR** Socially Interactive Robotics. 152
- SLAM** Simultaneous Localization and Mapping. 13, 38, 39, 42, 120, 127, 225
- SLS** Semantic Localization System. 120–122, 125, 127, 133
- SOD** Superficial Object Detector. 109, 112
- SVM** Support Vector Machines. 79, 80, 192–200
- TAUDL** Tracklet Association Unsupervised Deep Learning. 53

URDF Unified Robot Description Format. 90

VFD Voxelized Fractal Descriptor. 70, 73, 76, 83, 203, 239

VTK The Visualization Toolkit. 16, 228

WHO World Health Organization. 149

YOLO You Only Look Once. 15, 181, 182, 189, 192, 227



Universitat d'Alacant
Universidad de Alicante