

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



Control of a Robotic System with a Dexterous Hand for Detailed Tasks

Bruno Martins Santos

MESTRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES

Supervisor: Vítor Hugo Machado Oliveira Pinto

July 31, 2023

Reconhecimentos

The following people collaborated with the supervising team in carrying out this work: Francisco Manuel Barbosa Ribeiro.

This work was carried out with the support of SYSTEC and in the scope of Project *Digitalização da Arte Humana (Cibertoque)*, with reference POCI-01-0247-FEDER-072627, co-funded by FEDER, through COMPETE 2020.

Bruno Santos

Resumo

A automação de processos a nível industrial está atualmente a revolucionar de forma positiva o modo de funcionamento das indústrias. Isto deve-se às vantagens proporcionadas pela automação, nomeadamente, aumentos a nível de produtividade e a obtenção de um produto final com maior qualidade e repetibilidade.

Apesar de a automação se estar a expandir rapidamente pelas indústrias, em particular pela indústria automóvel, existem ainda diversos processos que são atualmente executados de forma manual. Isto deve-se principalmente a fatores como a elevada destreza manual requerida ou à necessidade de uma elevada sensibilidade ao toque, fatores estes que não são triviais de alcançar recorrendo a equipamento robótico.

Tendo isto em conta, esta dissertação insere-se no contexto do projeto “CIBERTOQUE”, que propõe a automação de dois dos processos mencionados anteriormente. Nomeadamente, o acondicionamento de cabos na estrutura do veículo e o controlo de qualidade da pintura automóvel, tarefas a realizar no contexto da empresa automóvel Stellantis. A abordagem proposta para a automação dos processos referidos envolve a utilização e respetivo controlo de uma mão robótica antropomórfica com elevada destreza acoplada a um braço robótico, fornecendo assim as condições necessárias à realização das tarefas propostas. Apesar do conjunto robótico mencionado, o foco irá incidir maioritariamente no controlo da mão robótica com o objetivo da replicação dos movimentos da mão humana.

Neste documento, são introduzidos e explicados conceitos fundamentais ao trabalho a realizar durante a dissertação, como o conceito de cinemática e sinergias. Adicionalmente, é apresentada a *Shadow Dexterous Hand*, a mão robótica escolhida para este projeto.

Subsequentemente, com a finalidade de controlo da mão robótica, diferentes algoritmos de remapeamento cinemático, incluindo DexPilot, BioIK e TeachNET, são introduzidos, explicados, implementados e comparados em detalhe. O objetivo principal deste estudo é determinar o algoritmo que melhor se enquadra com o contexto de aplicação específico relativo à dissertação.

Por fim, as soluções propostas para a realização das tarefas indicadas são apresentadas e implementadas experimentalmente no sistema robótico real.

O objetivo final desta dissertação é a demonstração da implementação bem sucedida das soluções desenvolvidas num ambiente relevante.

Palavras-chave: Shadow Dexterous Hand, UR5, manipulação robótica, remapeamento cinemático, indústria automóvel

Abstract

Processes automation at an industrial level is currently positively revolutionising the way industries operate. This is due to the advantages offered by automation, namely increases in productivity and the achievement of a final product with greater quality and repeatability.

Although automation is rapidly expanding throughout industries, in particular the automotive industry, there are still many processes that are currently being performed manually. This is mainly due to factors such as the high manual dexterity required or the need for high touch sensitivity, factors that are not trivial to achieve with robotic equipment.

Considering this, this dissertation is inserted in the context of the project “CIBERTOQUE”, which aims to automate two of the mentioned processes. Specifically, the wiring fitting and the automotive paint quality control tasks, inserted in the context of the automotive company Stellantis. The proposed approach for the automation of the aforementioned processes involves the utilisation and control of an anthropomorphic robotic hand with high dexterity coupled to a robotic arm, providing this way the necessary conditions to perform the proposed tasks. Despite the robotic set, the focus will centre on the control of the robotic hand, aiming to replicate the human hand’s movements.

In this document, fundamental concepts of the work to be developed during the dissertation, such as the concept of kinematics and synergies, are introduced and explained. Additionally, the Shadow Dexterous Hand, the chosen robotic hand for this research, is presented.

Subsequently, with the purpose of robotic hand control, various kinematic retargeting algorithms, including DexPilot, BioIK and TeachNET, are comprehensively introduced, explained, implemented, and compared. The primary objective is to determine the most suitable algorithm for the specific application context of this dissertation.

Lastly, the proposed solutions for the designated tasks are presented and experimentally implemented on the physical robotic system.

The ultimate objective of this dissertation is the demonstration of the successful implementation of the developed solutions in a relevant environment.

Keywords: Shadow Dexterous Hand, UR5, robotic manipulation, kinematic retargeting, automotive industry

Agradecimentos

Apesar de a dissertação ser um trabalho individual, a sua realização não seria possível sem a orientação, apoio e importantes contribuições de diversas pessoas. De seguida, gostaria de expressar os meus agradecimentos a todos aqueles que, de alguma forma, contribuíram para a concretização desta dissertação.

Em primeiro lugar, gostaria de agradecer ao meu orientador, Professor Vítor Pinto, por todo o seu apoio e dedicação, especialmente durante esta parte final do meu percurso académico. Gostaria de agradecer também o incentivo constantemente em relação à investigação, proporcionando-me oportunidades enriquecedoras que me forçaram a sair da zona de conforto e, sem dúvida alguma, a evoluir como estudante e pessoa. Todo o seu apoio e orientações contribuíram de forma valiosa para a qualidade do trabalho realizado.

Agradeço também ao Engenheiro Francisco Ribeiro, pela sua preciosa colaboração e ajuda durante todo o período de dissertação. Agradeço pelo tempo despendido em trocas de ideias e sobretudo no auxílio com o vasto repositório da Shadow Hand, tanto na sua compreensão como na resolução de erros e incompatibilidades que surgiam constantemente. A sua experiência e auxílio foram cruciais para a progressão do projeto realizado.

Gostaria também de agradecer a todos os elementos da equipa do projeto CIBERTOQUE, no qual se enquadra a dissertação, nomeadamente ao Professor Vítor Pinto, Engenheiro Francisco Ribeiro e Tiago Correia. A experiência, conhecimento e contribuição de cada um dos elementos foi indispensável à concretização do projeto.

Queria agradecer também a todos os participantes que contribuíram para o teste experimental realizado no contexto da dissertação. A colaboração de todos foi determinante para a realização do estudo e obtenção de resultados significativos. Agradeço pela disponibilidade e disposição pela participação do estudo.

Por último, mas não menos importante, deixo um agradecimento especial à minha família e amigos que me acompanharam durante esta jornada académica. O amor, amizade, encorajamento e apoio incondicional foram fatores determinantes para a minha motivação e bem-estar. Agradeço por terem sempre acreditado em mim.

A todos, o meu Muito Obrigado!

Bruno Santos

*“The greatest danger for most of us
is not that our aim is too high and we miss it,
but that it is too low and we reach it.”*

Michelangelo Buonarroti

Contents

Reconhecimentos	i
Resumo	iii
Abstract	v
Agradecimentos	vii
Abbreviations and Symbols	xix
1 Introduction	1
1.1 Context	1
1.2 Motivation	1
1.3 Objectives	2
1.4 Document Structure	3
2 Background	5
2.1 Kinematics	5
2.2 Synergies	6
2.3 Shadow Dexterous Hand	7
2.3.1 Sensing	8
2.3.2 Control	9
2.3.3 DIGIT Sensor	9
3 Literature Review	11
3.1 Data Acquisition	11
3.1.1 Databases	11
3.1.2 Data Gloves	12
3.1.3 Vision-Based Techniques	12
3.2 Dimensionality Reduction	14
3.2.1 Principal Component Analysis	14
3.2.2 Non-linear Autoencoder Networks	14
3.3 Kinematic Retargeting	16
3.3.1 DexPilot	16
3.3.2 BioIK	17
3.3.3 Hand Embodiment	18
3.3.4 TeachNET	19
3.4 Learning from Demonstration	20
3.4.1 Gaussian Mixture Regression	20

3.4.2	Dynamic Movement Primitives	20
3.4.3	Concept Constrained Learning from Demonstration	22
3.4.4	Artificial Neural Networks	23
3.5	Automotive Painting Quality Control	24
4	Implementation	27
4.1	Data Acquisition	27
4.2	Teleoperation	28
4.2.1	Direct Joints Mapping	28
4.2.2	DexPilot	31
4.2.3	BioIK	36
4.2.4	Hand Embodiment	37
4.2.5	TeachNET	38
5	Shadow Hand Control Results	41
5.1	MuJoCo Simulation versus Real-World Control	41
5.1.1	Joint Coupling	41
5.2	Teleoperation	45
5.2.1	Direct Joints Mapping	45
5.2.2	DexPilot	46
5.2.3	BioIK	48
5.2.4	TeachNET	50
5.3	Algorithms Comparison	52
6	Implementation of Wiring Fitting and Paint Quality Inspection Tasks	57
6.1	Wiring Fitting	57
6.1.1	Developed Algorithm	57
6.1.2	Clip Adapter	61
6.1.3	Experimental Implementation	62
6.2	Automotive Painting Quality Control	65
7	Conclusions	67
7.1	Future Work	68
7.2	Scientific Contributions	69
A	Kinematic Retargeting Algorithms Test Results	71
	References	83

List of Figures

2.1	UR5 manipulator [8]	6
2.2	UR5 kinematic model [8]	6
2.3	Shadow Hand [6]	8
2.4	Shadow Hand kinematic model [6]	8
2.5	DIGITs coupled to a robotic hand [16]	10
2.6	Raw images captured by DIGIT sensor [16]	10
3.1	Cyber Glove II [19]	12
3.2	Sensorial glove prototype [20]	12
3.3	Average 3D distance error from Ground Truth (GT) in millimetres [21]	13
3.4	3D hand skeletal and 3D virtual hand [25]	13
3.5	Hand pose reconstructions comparison (utilizing 3 PC) [28]	15
3.6	Success rate of tasks across two pilots run over five successive trials without reset [32]	17
3.7	Kinematic retargeting results between the human hand model and the Allegro hand model [32]	17
3.8	Shadow Hand experiment performed in Gazebo simulation and in reality [34]	18
3.9	TeachNet architecture [37]	19
3.10	Successful teleoperation results [37]	19
3.11	Virtual Spring Framework represented in an object grasped by four fingers	22
4.1	OpenPose hand output format (from Openpose documentation ¹)	29
4.2	Hand keypoints before (in blue) and after (in white) the application of the median filter	30
4.3	Program architecture represented in a UML sequence diagram	30
4.4	Space vectors utilised in DexPilot for both the human hand model and Allegro hand [32]	31
4.5	Space vectors adapted for Shadow Hand	31
4.6	Thumb kinematic model	33
4.7	Little finger kinematic model	33
4.8	Remaining fingers kinematic model	33
4.9	Program architecture represented in a UML sequence diagram	36
4.10	Keypoints before (in blue) and after (in white) the application of robotic hand mapping	37
4.11	Hand Embodiment algorithm steps: human hand markers (left), MANO model (middle) and Shadow Hand mapping (right) [35]	38
4.12	ZED (left) and Microsoft Kinect v2 (right) depth images	39
4.13	TeachNET filtered image (acquired with Microsoft Kinect v2)	39

4.14	Program architecture represented in a UML sequence diagram	40
5.1	Human Hand position for robotic hand replication reference	42
5.2	Hand pose replication in simulation (left) and in reality (right) by DexPilot before taking into account the joints coupling	42
5.3	Hand pose replication in simulation (left) and in reality (right) by DexPilot after taking into account the joints coupling	42
5.4	Hand pose replication in simulation (left) and in reality (right) by BioIK before taking into account the joints coupling	43
5.5	Hand pose replication in simulation (left) and in reality (right) by BioIK after taking into account the joints coupling	43
5.6	Joints coupling test qualitative results in MuJoCo and in the real Shadow Hand .	44
5.7	Clips utilised for the wiring fitting task	45
5.8	Direct joints mapping algorithm test results	46
5.9	DexPilot: experimental results	47
5.10	DexPilot: pinching pose repeatability	47
5.11	DexPilot: distance between thumb tip and forefinger tip	48
5.12	BioIK: experimental results	49
5.13	BioIK: pinching pose repeatability	49
5.14	BioIK: distance between thumb tip and forefinger tip	50
5.15	BioIK: distances between human and robotic fingertips positions	50
5.16	TeachNET process: human hand pose, Kinect depth image, TeachNET segmented image, robotic hand pose, from left to right, respectively	51
5.17	TeachNET: experimental results	51
5.18	TeachNET: pinching pose repeatability	52
5.19	Performance of each participant by algorithm	54
5.20	Performance of each participant by clip	54
5.21	Performance of each participant by attempt	54
5.22	Global performance of each participant	54
5.23	Global performance by algorithm	54
5.24	Global performance by clip	54
5.25	Global performance by attempt	54
5.26	Global maximum score attempts by algorithm	55
5.27	Global maximum score attempts by clip	55
5.28	Global maximum score attempts by attempt	55
5.29	OpenPose misdetections (left image for reference)	55
5.30	Successful grasps from tests. The rows, from top to bottom, correspond to the DexPilot, BioIK and TeachNET algorithms, respectively. The left to right columns correspond to clips 1, 2 and 3, respectively.	56
6.1	UML State Machine Diagram of the developed algorithm	57
6.2	Joint Slider plugin	58
6.3	Graphical Interface of the algorithm	59
6.4	Hand Positions for states “Hand_Open”, “Start_Pinching_Position” and “Pinching_Position”, respectively	59
6.5	Unsuccessful clipping attempt with the developed algorithm (left, top and right viewpoints, respectively)	60
6.6	Clenched fist hand positions for states “Start_Pinching_Position” and “Pinching_Position”, respectively	60

6.7	Clenched fist hand position holding the clip	60
6.8	Clenched fist hand position performing the clipping task	61
6.9	Clip Adapter: 3D model (left side) and 3D printed result (right side)	61
6.10	Shadow Hand grasping the clip with the adapter from different angles	62
6.11	Clipping task performance utilising the clip adapter	62
6.12	Numerically identified clips and corresponding clipping locations	63
6.13	Wiring fitting task: steps 1 to 6, from left to right, respectively	64
6.14	Automotive painting quality inspection process	66

List of Tables

3.1	First and second tasks performance results [43]	23
4.1	Thumb DH parameters	34
4.2	Little finger DH parameters	34
4.3	Remaining fingers DH parameters	34
5.1	Joints coupling test quantitative results in MuJoCo and in the real Shadow Hand	44
A.1	Participant 1 score for each attempt utilizing TeachNET algorithm	72
A.2	Participant 1 score for each attempt utilizing BioIK algorithm	72
A.3	Participant 1 score for each attempt utilizing DexPilot algorithm	72
A.4	Participant 2 score for each attempt utilizing TeachNET algorithm	73
A.5	Participant 2 score for each attempt utilizing BioIK algorithm	73
A.6	Participant 2 score for each attempt utilizing DexPilot algorithm	73
A.7	Participant 3 score for each attempt utilizing TeachNET algorithm	74
A.8	Participant 3 score for each attempt utilizing BioIK algorithm	74
A.9	Participant 3 score for each attempt utilizing DexPilot algorithm	74
A.10	Participant 4 score for each attempt utilizing TeachNET algorithm	75
A.11	Participant 4 score for each attempt utilizing BioIK algorithm	75
A.12	Participant 4 score for each attempt utilizing DexPilot algorithm	75
A.13	Participant 5 score for each attempt utilizing TeachNET algorithm	76
A.14	Participant 5 score for each attempt utilizing BioIK algorithm	76
A.15	Participant 5 score for each attempt utilizing DexPilot algorithm	76
A.16	Participant 6 score for each attempt utilizing TeachNET algorithm	77
A.17	Participant 6 score for each attempt utilizing BioIK algorithm	77
A.18	Participant 6 score for each attempt utilizing DexPilot algorithm	77
A.19	Participant 7 score for each attempt utilizing TeachNET algorithm	78
A.20	Participant 7 score for each attempt utilizing BioIK algorithm	78
A.21	Participant 7 score for each attempt utilizing DexPilot algorithm	78
A.22	Participant 8 score for each attempt utilizing TeachNET algorithm	79
A.23	Participant 8 score for each attempt utilizing BioIK algorithm	79
A.24	Participant 8 score for each attempt utilizing DexPilot algorithm	79
A.25	Participant 9 score for each attempt utilizing TeachNET algorithm	80
A.26	Participant 9 score for each attempt utilizing BioIK algorithm	80
A.27	Participant 9 score for each attempt utilizing DexPilot algorithm	80
A.28	Participant 10 score for each attempt utilizing TeachNET algorithm	81
A.29	Participant 10 score for each attempt utilizing BioIK algorithm	81
A.30	Participant 10 score for each attempt utilizing DexPilot algorithm	81

Abbreviations and Symbols

2D	Two-Dimensional
3D	Three-Dimensional
A2J	Anchor to Joint
Adam	Adaptive Moment Estimation
CC-LfD	Concept Constrained-Learning from Demonstration
cm	centimetres
CPU	Central Processing Unit
DCGAN	Deep Convolutional Generative Adversarial Network
DH	Denavit-Hartenberg
DMPs	Dynamic Movement Primitives
DoF	Degrees of Freedom
DR	Dimensionality Reduction
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
GPU	Graphics Processing Unit
GT	Ground Truth
HD	High-Definition
IMU	Inertial Measurement Unit
KPCA	Kernel Principal Component Analysis
LbD	Learning by Demonstration
LED	Light-Emitting Diode
LfD	Learning from Demonstration
MANO	hand Model with Articulated and Non-rigid defOrmations
MSE	Mean Squared Error
MuJoCo	Multi-Joint dynamics with Contact
nAEN	non-linear Autoencoder Network
Ninapro	Non Invasive Adaptive Prosthetics
PC	Principal Components
PCA	Principal Components Analysis
PSTs	Pressure Sensor Tactiles
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RGB	Red Green Blue
RGB-D	Red Green Blue-Depth
RL	Reinforcement Learning
ROS	Robot Operating System
SELU	Scaled Exponential Linear Units
SLSQP	Sequential Least Squares Programming

ST-A2J	Spatial-Temporal Anchor to Joint
Tanh	Hyperbolic Tangent
TF	Transform System
ToF	Time-of-Flight
UML	Unified Modeling Language
VAF	Variance Accounted For
VSF	Virtual Springs Framework
YOLO	You Only Look Once

Chapter 1

Introduction

This chapter introduces the work to be carried out during the dissertation. It presents the context in which it is inserted, the reasons that lead to its realisation, and the intended objectives to be achieved. Finally, a description of the structure of the document is given.

1.1 Context

Automation of industrial processes is something that has been seen over the years, and that is dramatically changing the way industry works. Since the 1960s, with the improvement of technology, industrial-level processes performed until that time by human operators have gradually been replaced by industrial robots [1].

Initially, industrial automation consisted of simple conveyors moving parts through a manufacturing line; however, as time went by, robots quickly evolved and spread rapidly worldwide as a result of manufacturers' competitiveness [2].

The automotive industry sector, which is in the scope of this dissertation, was, in fact, one of the sectors where there was one of the largest implementations of robots into production, with 33% of total supply robots in 2019 [3]. In this industry sector, for now, some partial or full robotised operations include: casting¹, grinding and polishing, cutting, assembly, painting and welding [2], but there are still tasks that cannot be automated. The main reasons are that these tasks require a high degree of manual dexterity, or the cost to automate the task does not compensate for the increase in productivity.

1.2 Motivation

Industrial automation has become imperative for companies striving to attain competitiveness in the contemporary marketplace, owing to its inherent advantages [3]. In fact, papers about this topic have been published as early as 1984 [4], indicating its enduring significance. One of the

¹casting: “process of forming products through melting engineering materials and pouring it in preformatted spaces called moulds” [2]

primary benefits undoubtedly lies in the heightened productivity that it engenders [5]. This happens due to the inherent swiftness of robots compared to human operators, facilitating continuous operations. Furthermore, with less need for human operators, labour costs and human errors will decrease, ultimately resulting in better product quality.

As elucidated, industrial automation is an expansive domain with vast potential, destined to assume a crucial role in companies' future. Consequently, it is essential for industries to concentrate efforts on optimising the level of industrial automation. In the specific case of the automotive industry, which is already predominantly automated, maximising automation involves dealing with certain manual handling operations that continue to pose challenges. Such tasks typically necessitate heightened touch sensitivity, like quality inspection tasks, or involve complex hand movements that are nontrivial to replicate using robotic systems.

The automation of the aforementioned tasks offers compelling benefits. Presently, these tasks rely on human involvement, making it exceedingly challenging to ensure consistent work methodologies and criteria among different individuals, resulting in slight variations in the final product. However, by employing robots to perform these tasks, the execution would be optimised to achieve the best possible results. This not only enhances product quality but also enhances repeatability to a greater extent.

1.3 Objectives

This dissertation is situated within the context of the project “Digitalização da Arte Humana” (CIBERTOQUE), translated as "Digitization of Human Art". Consequently, the objectives of this dissertation align naturally with the project's objectives.

The primary purpose is to develop a robotic system that dexterously replicates human movements. The application of this system will be directed towards two specific tasks within Stellantis, an automotive company. These tasks, namely wire fitting and automotive paint quality inspection, are presently carried out manually by human operators.

To achieve the objectives outlined and address the two proposed tasks, the proposed approach involves controlling an anthropomorphic² robotic hand [6] coupled with a collaborative robotic arm [7]. The robotic set must possess adequate dexterity to execute the wire fitting operation, which involves securely fastening cables to the car structure. Additionally, it must exhibit sufficient sensitivity to conduct thorough inspections of vehicle paint quality.

The final objective is to demonstrate the successful implementation of the developed solutions in a relevant environment. Thus, contributing to the increase of automation level in the automotive industry, achieving all the benefits referred to, such as superior productivity, efficiency and quality.

²anthropomorphic: “treating animals, gods, or objects as if they are human in appearance, character, and behaviour” (from Cambridge Dictionary)

1.4 Document Structure

This document comprises six subsequent chapters. Chapter 2 provides a concise introduction to and explication of fundamental theoretical concepts pertinent to the scope of the dissertation, including kinematics and synergies. This chapter also introduces the hardware to be utilised. In Chapter 3, a state-of-art study will be conducted on several topics important to the dissertation work's development. These themes encompass data acquisition, kinematic retargeting, dimensionality reduction, learning from demonstration and automotive painting quality control. Chapter 4 delineates the implementation of methodologies and algorithms employed for controlling the robotic system. In Chapter 5, are presented the outcomes attained through the utilisation of the aforementioned algorithms. Chapter 6 presents the specific application case of the developed algorithms to the wiring fitting and automotive paint quality inspection tasks. Lastly, Chapter 7 encapsulates the conclusions, suggestions for future research and scientific contributions derived from the execution of the dissertation's work.

Chapter 2

Background

This chapter provides contextual information and fundamental theoretical knowledge pertaining to the specific engineering domain addressed in the dissertation. It aims to present and provide concise explanations of essential theoretical concepts. Furthermore, the hardware that will be utilised in the dissertation's work will be introduced in this chapter.

2.1 Kinematics

A comprehensive understanding of robotic manipulators, such as robotic arms and hands, requires consideration of kinematics models. Kinematics, in essence, is the scientific area concerned with the study of robot movements, thus dealing with variables such as position, velocity, and acceleration.

The kinematic model of a manipulator consists of its links, connected by joints predominantly of prismatic or revolute nature. In two dimensions (2D), prismatic joints are visually depicted as squares, while in three dimensions (3D), they are represented as cubes. These joints correspond to translation movements. Conversely, revolute joints are visually represented in 2D as circles or rectangles (depending on the perspective) and in 3D as cylinders, denoting rotational movements, as the name implies.

An additional noteworthy aspect to consider is the concept of degrees of freedom (DoF). The DoF directly correlates with the motion capability of a manipulator and is determined by the number of joints present in the kinematic model. Specifically, each prismatic or revolute joint independently actionable corresponds to one cumulative degree of freedom.

It is from the kinematic model that the position and orientation of the end effector are calculated, taking into account the values of each joint parameter, a process referred to as forward kinematics. The opposite, i.e., calculating the values of each joint parameter taking into account the desired position and orientation of the end effector, can also be done, and it is called inverse kinematics. In addition, other quantities, such as forces and torques at the joints, are also calculated using kinematics. This substantiates the significance of kinematics as a crucial and pertinent topic within the domain of study.

To enhance practical comprehension of the aforementioned concepts, a concise analysis of the kinematic model of the “UR5” robotic arm [8], shown in Figure 2.2, will be undertaken.

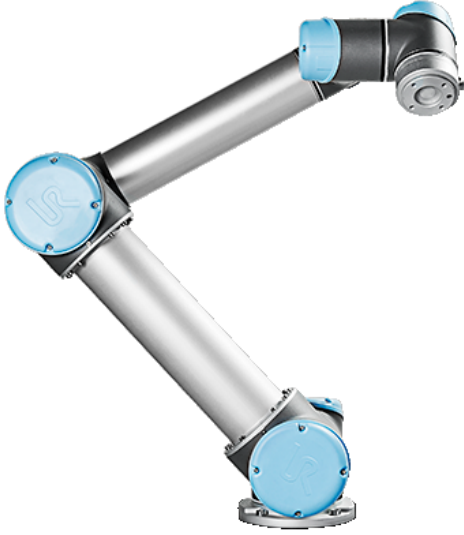


Figure 2.1: UR5 manipulator [8]

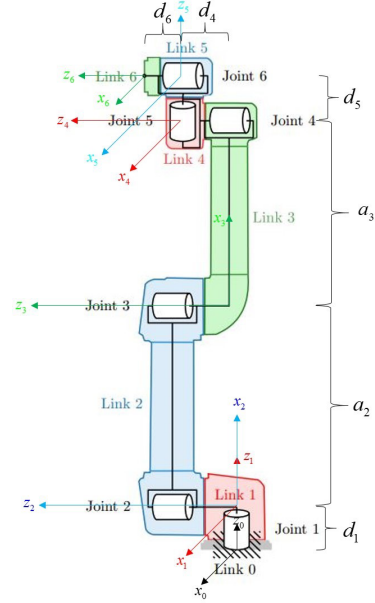


Figure 2.2: UR5 kinematic model [8]

The kinematic model depicted in Figure 2.2, annotated according to the modified Denavit-Hartenberg convention, exclusively comprises revolute joints, indicating the exclusive presence of rotational motion in this manipulator. Counting the number of joints leads to the intuitive conclusion that the UR5 manipulator possesses a total of 6 DoF. Furthermore, a noteworthy characteristic of this manipulator is that the last three joints do not act as a coincidental wrist, resulting in rotational movements totally independent of each other. Consequently, all of its six joints contribute to the transformational and rotational movements of the end effector [8]. For reference, in Figure 2.2, the origin of the end effector is determined by the intersection of the x_6 and z_6 axes.

2.2 Synergies

Throughout the years, robotic hands have undergone progressive advancements, gradually approaching the level of complexity exhibited by the human hand. This evolution has consisted of an increase in the number of fingers, as well as an increase in the number of associated joints and, consequently, DoF. Such developments are advantageous as they enable robots to attain dexterity that closely resembles that of the human hand; however, in practice, it is necessary to develop a control algorithm capable of providing this high level of dexterity.

At first, it would be expected that a large number of DoF intrinsic to the human hand coupled with its high dexterity would imply an exceedingly complex control. However, investigations into hand movements, contact forces and muscle activities have unveiled a correlation among DoF [9]. Consequently, the total number of final variables to control will be lower than the number of DoF,

thereby significantly diminishing the complexity of the control algorithm. These final variables are termed synergies.

Quantitative results were obtained in [10]. In this article, the authors took into account hand kinematic data of 77 subjects from Ninapro (Non Invasive Adaptive Prosthetics) database [11], performing up to 20 hand grasps. The acquired data were subsequently analysed using the Principal Components Analysis (PCA) [12] approach. As a result, a total of twelve synergies were identified, effectively characterising over 80% of the overall variation observed in hand movements. Moreover, it was determined that merely three of these synergies accounted for more than half of the total variance observed.

Although utilising the concept of synergies to reduce the dimensionality space of hand kinematics demonstrates promising results, the application of synergistic control to perform complex manipulation tasks is still an open issue [13].

2.3 Shadow Dexterous Hand

This section will present the principal physical component of the dissertation work, the robotic hand.

Given the requirement for the robotic hand to perform tasks demanding precise and dexterous movements, it is imperative to ensure that the robot possesses the requisite capabilities. To address this need, it will be utilised the Shadow Dexterous Hand [6]. Shadow Hand is a humanoid robotic hand that provides high flexibility of movements due to its 24 joints, specifically engineered to replicate, with the utmost fidelity, the kinematics and precision of the human hand.

Regarding its dimensions, the Hand has been meticulously designed to resemble the proportions of a typical male hand closely. Although all fingers possess identical sizes, the knuckles have been strategically offset to ensure that the fingertips are located in a human-like region.

Shadow Dexterous Hand has been previously employed in research contexts such as manipulation and industry quality control, which aligns with the context of this dissertation. This adds a degree of confidence regarding its suitability for the dissertation context. All the referred information is present in the manual “Shadow Dexterous Hand Technical Specifications” [6].

Figure 2.4 shows the kinematic model of the Shadow Dexterous Hand. Upon careful examination, it becomes evident that the hand solely comprises revolute joints.

Analysing the thumb, it is observed that it comprises a total of five joints, corresponding to five DoF, as discussed in Section 2.1. Conversely, the remaining fingers possess four joints each; however, their effective degrees of freedom are reduced to three. This reduction is attributed to the coupling of the joints defined in Figure 2.4 as *Finger 1* and *Finger 2*, i.e., these two joints cannot be moved independently. Under the absence of external forces, the first joint only moves when the second joint reaches its maximum range (fully closed). To account for this behaviour, a fictitious Joint 0 is introduced, resulting in the following model for these joints:



Figure 2.3: Shadow Hand [6]

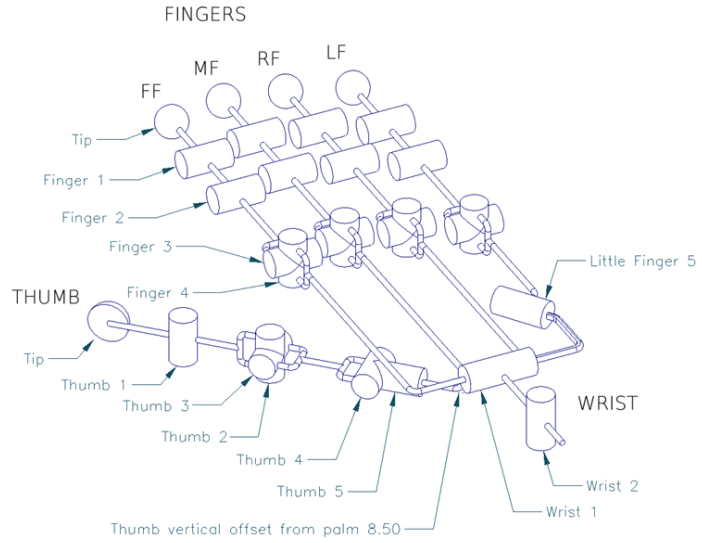


Figure 2.4: Shadow Hand kinematic model [6]

$$\begin{aligned} \text{Joint 1} &= \max(0^\circ, \text{Joint 0} - 90^\circ) \\ \text{Joint 2} &= \min(90^\circ, \text{Joint 0}) \end{aligned} \quad (2.1)$$

However, when external forces are present, for example, in a grasp, the behaviour of these joints is not deterministic and is merely defined by:

$$\text{Joint 1} \leq \text{Joint 2} \quad (2.2)$$

This occurs due to a single tendon's shared control of the two mentioned joints.

Therefore, by summing the DoF of all the fingers with the remaining joints in the model, a total of 20 DoF are obtained for a total of 24 joints. In practical terms, individually controlling these 20 variables would entail a highly complex control system. Therefore, there arises a necessity to decrease the DoF number. This is precisely where synergies, as discussed in Section 2.2, play a significant role.

2.3.1 Sensing

Shadow Dexterous Hand is equipped with a large number of sensors [6]. It includes position sensors that provide precise measurements of the rotation of each joint with a resolution of 0.2 degrees. Force sensors are responsible for measuring the force in each pair of tendons. Temperature and current sensors ensure that the Shadow is operated safely and reliably. And lastly, the Pressure Sensor Tactiles (PSTs), a single region sensor with high sensitivity located at each fingertip.

Once again, all the aforementioned information is present in the manual “Shadow Dexterous Hand Technical Specifications” [6].

2.3.2 Control

The Shadow Hand will be controlled by utilising the Hand Commander software, specifically developed for operation within the Robot Operating System (ROS) environment. Hand Commander provides a high-level interface for controlling the Shadow, enabling users to transmit commands and receive feedback from the Shadow Hand.

To command the Shadow Hand, the user will specify the desired angle for each one of the Hand joints, which totals 24 angles. From the Shadow Hand, it is possible to get various information, such as joints positions and effort and tactile sensors state.

Since the Shadow Dexterous Hand is a piece of expensive equipment, in order to mitigate the risk of premature wear and potential damage, which can occur due to, for example, collisions between fingers, the control algorithms will be initially tested in a simulation environment, specifically in Multi-Joint dynamics with Contact (MuJoCo) [14]. MuJoCo is a versatile physics engine that aims to facilitate research and development in various areas, including robotics, that require fast and accurate simulations. The selection of MuJoCo as the preferred simulator arose from work conducted by Francisco Ribeiro in his Master’s Dissertation [15], responsible also for the integration between the software provided by Shadow Hand company and the MuJoCo simulator, thus allowing the control of the robotic hand in a simulation environment.

2.3.3 DIGIT Sensor

One of the designated tasks to be accomplished by the hand resides in the automotive paint quality inspection. Currently, this task is performed by a human operator who assesses the quality of the paint by running his hand over the surface, identifying minor irregularities and defects through tactile feedback.

The primary objective of this dissertation is to replicate human movements; nonetheless, incorporating a component into the system that actively verifies the quality of the paint would be valuable. With this objective in mind, the proposed plan is to utilise “DIGIT” sensors [16] attached to the fingers of Shadow Hand in an identical way to the one shown in Figure 2.5. These additional sensors are anticipated to enhance the hand’s capability to perceive and accurately assess the vehicle paint’s quality.

DIGIT is a tactile sensor explicitly designed for attachment to robotic hand fingertips, offering advantages such as affordability, compactness, and high resolution. The objective of this coupling is to accurately estimate contact forces, thus increasing the dexterity level of the robotic hand.

For the purpose of paint quality inspection, the primary advantage of utilising these sensors does not lie in their contribution to dexterity enhancement but rather in their vision-based capabilities. Each sensor is equipped with a camera and three integrated lights (red, green and blue) that

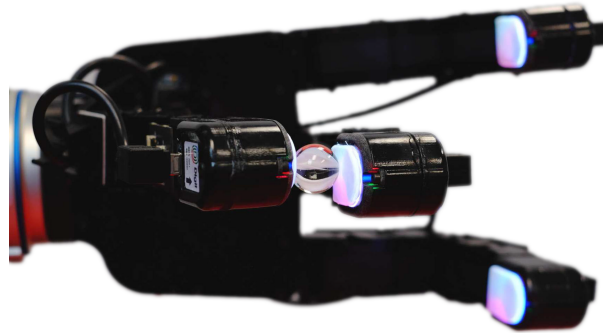


Figure 2.5: DIGITs coupled to a robotic hand [16]

provide essential information to the good results of the quality control task by detecting imperfections almost invisible to the human eye or touch.

To exemplify the visual capabilities of the DIGIT sensor, Figure 2.6 displays several objects alongside the corresponding detailed images captured by the DIGIT. Notably, the images captured by DIGIT demonstrate its ability to acquire the texture of objects with good resolution. When combined with computer vision techniques, this characteristic becomes pivotal to ensuring robust and accurate quality control.

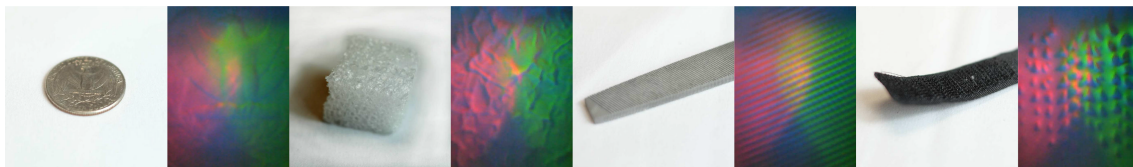


Figure 2.6: Raw images captured by DIGIT sensor [16]

In addition to the aforementioned advantages, the DIGIT sensor is open-source, which means that modifications can be made throughout the dissertation, if deemed necessary, to both the software and hardware components, enabling a better adaption to the specific application.

Chapter 3

Literature Review

In this chapter, important subjects will be addressed in the context of the dissertation. For each of the topics, the respective literature review will be presented.

3.1 Data Acquisition

One of the main approaches to define the robotic hand synergies consists of mapping synergies from humans to robots [13]. This approach works especially well when the robotic hand's kinematic model is similar with the human hand, which, as seen in Section 2.1, is the case. Furthermore, the utilisation of data acquired from the human hand allows the exploitation of the control mechanisms of the human brain, which is the result of thousands of years of evolution [13].

In order to put this method into practice, it is first necessary to collect kinematic data from the human hand. This chapter aims to explore these data-collecting techniques.

3.1.1 Databases

One of the less complex techniques consists of utilising already acquired data from a database.

Ninapro database, already referred to in Section 2.2, is one of the best examples in this scope. This project started in 2011 with the main aim of increasing the dexterity level of hand prostheses by collecting a large amount of hand kinematic data. Although this was the main goal, from the beginning of building this database, industrial applications were also taken into account [17].

In the following year, 2012, the first version of the Ninapro database was introduced, containing data from 27 people, each performing a total of 52 movements. In this first version, it was already foreseen that Ninapro would represent a real challenge for the state of the art. [18]

Since this first version, new versions have been released, such as “*Ninapro DB9*”. This dataset stands out by including data from 77 subjects and 40 movements, thus resulting in one of the biggest kinematic datasets available [19].

The utilisation of an existing database, such as Ninapro, has the advantage that it is unnecessary to collect the data, as mentioned at the beginning of this section. However, due to the

large amount of data from so many different hand movements, the generated synergies may not fit properly for such specific movements, as is the case of wire fitting.

3.1.2 Data Gloves

Another technique involves the utilisation of a glove to measure the kinematic hand data.

One example of this type of equipment is the CyberGlove II, shown in Figure 3.1. This was, in fact, the glove used to obtain the data for the database referred to in Section 3.1.1. This glove is equipped with 22 sensors responsible for real-time joint-angle data acquisition [19]. One of the significant advantages resulting from the utilisation of this particular glove is the great accuracy of the collected data. On the other hand, a considerable disadvantage resides in the high cost of this equipment.

Another example includes the sensorial glove prototype developed by Tiago Cerqueira *et al.* [20], shown in Figure 3.2. This equipment consists of a regular wool glove, with 12 Inertial Measurement Units (IMUs) attached at strategic positions to acquire the hand's position. This design allows for a very significant cost reduction while providing satisfactory results.



Figure 3.1: Cyber Glove II [19]

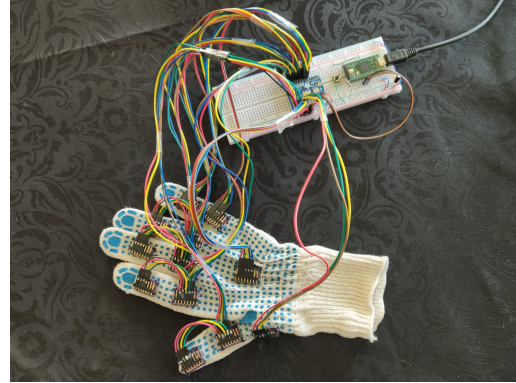


Figure 3.2: Sensorial glove prototype [20]

The utilisation of data gloves allows for greater flexibility in capturing kinematic hand data since they allow for the acquisition of hand movements that are actually relevant to the intended task. The disadvantages include either the high cost or an inferior accuracy of data.

3.1.3 Vision-Based Techniques

Lastly, there are vision-based techniques. These techniques usually resort to neural networks to predict the human hand position, having as input hand images, usually with depth.

One example of the implementation of this technique is in [21]. In this article, it is utilised an Intel RealSense SR300 RGB-D (Red Green Blue - Depth) camera that, as the name implies, simultaneously captures colour and depth video.

For data processing, an Anchor-to-Joint Network (A2J) [22]. This network corresponds to the state-of-the-art of estimating 3D hand position. By testing A2J, it was found to perform well for single independent frames but not in videos due to the inconsistency of hand keypoints detected

across consecutive frames. As a solution to this problem, the Spatial-Temporal A2J (ST-A2J) was proposed. This model has several consecutive depth frames as input instead of treating each frame individually. After that, as refinement, some heuristics were also taken into account, such as that the joints' angle does not vary significantly in consecutive frames and that the distance between keypoints, this is, the hand bones length, is fixed. In order to apply the Refined ST-A2J algorithm, the region of interest (the hand) was firstly defined using a YOLO (You Only Look Once) V3 [23] network. The dataset used for training and testing was HANDS 2017 [24].

After all the improvements were applied, the results significantly exceeded those obtained with the A2J network alone, as seen in Figure 3.3, thus proving the efficiency of the approach followed in this paper.

A similar approach was taken in [25], but using monocular RGB images with no depth as input. In this case, the goal was to detect 21 hand keypoints and to reconstruct a 3D virtual hand, as seen in Figure 3.4. In order to increase accuracy, three datasets were used for training. The method used consists of three modules. The first step consists of palm detection since the palm is immune to the majority of possible occlusions, thus improving the algorithm's robustness. The subsequent steps are related to the 3D estimation of the hand pose and the respective rotational 3D mapping.

The developed network was tested in two datasets, obtaining an average accuracy of 88.85%. These results show a significant improvement over the state-of-the-art networks.

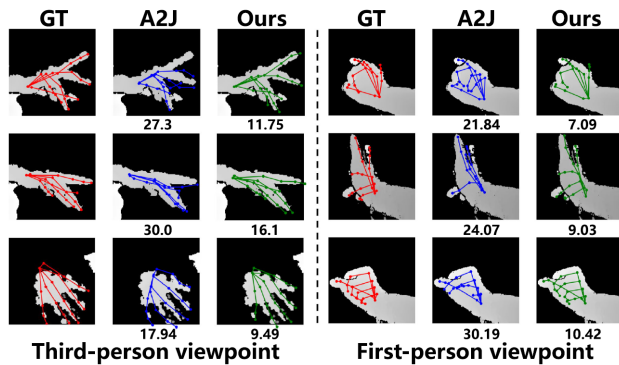


Figure 3.3: Average 3D distance error from Ground Truth (GT) in millimetres [21]

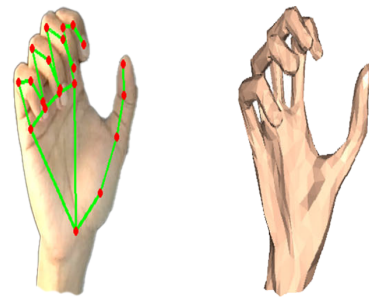


Figure 3.4: 3D hand skeletal and 3D virtual hand [25]

Concluding, the utilisation of vision-based techniques comprises the flexibility of data glove methods and provides more freedom, with the hand not being confined to a glove. However, general vision-based typical problems can occur, such as occlusion, depth ambiguity perceptions and in some cases, the background can be confused with the hand. Furthermore, this is a complex technique to implement.

3.2 Dimensionality Reduction

As referred to in Section 2.2, the synergies concept corresponds to the space Dimensionality Reduction (DR) of hand kinematics. In this chapter, DR techniques are introduced, explored and compared in terms of performance.

3.2.1 Principal Component Analysis

One of the classical DR methods is PCA, used, for instance, in [10], as referred in Section 2.2. This method is based on the data covariance matrix. The Principal Components (PC) are obtained by calculating the eigenvectors and eigenvalues of the covariance matrix. The first component is the one that describes the most data variation, being the second component the one that represents the second most data variation, and so on. Quickly recalling the results, for Ninapro database [11], twelve principal components account for more than 80% of the overall variation.

Another example of the utilisation of PCA is in [26] (2022). In this article, PCA is used in order to synergistic control a soft robotic hand. A dataset was first created with a total of 33 different human grasps, and each is repeated three times. Then PCA was applied to the dataset. PCA results showed that only two PC counted for 89,9% of dataset variation. According to this result, the soft robotic hand was then controlled using only two inputs. Of the 33 human grasps, 30 were satisfactorily replicated by the robotic hand, achieving a success rate of around 90%.

3.2.2 Non-linear Autoencoder Networks

PCA is a linear DR method, i.e., it only works properly with linear data. Considering the hand structure, the assumption that the hand kinematic is linear is not absolutely correct. So an alternative is exploring non-linear DR techniques that may perform better.

In article [27], one of these methods was explored. Firstly, a dataset was created, extracting kinematic data from 10 individuals, each performing a total of 30 different gestures. The data acquisition method was a data glove (Virtual Motion Labs, Dallas, TX, USA). This dataset was then divided into 80% and 20% parts to the training and testing phases, respectively.

Two methods were utilised for the data analysis: the PCA and a non-linear Autoencoder Network (nAEN). An autoencoder network is a symmetric network in which the middle layer is the bottleneck. The number of columns of this specific layer corresponds to the number of desirable components, similar to PC in PCA. This network is trained to have the same outputs as the given inputs, hence the term autoencoder. In the end, the left part of the network acts as an encoder and the right part as a decoder.

In [27], the nAEN architecture included three hidden layers. In the first and last hidden layers, it was used Hyperbolic Tangent (Tanh), a non-linear activation function. Other activation functions were tried, such as sigmoid and Rectified Linear Unit (ReLU); however, the performance was similar in all cases. The optimiser utilised was Adaptive Moment Estimation (Adam). The

performance metric utilised was Variance Accounted For (VAF) which measures the difference between predicted and original values.

The results were as expected. The nAEN outperformed PCA for any number of components considered. With only two components, the average VAF was 94% and 78% for nAEN and PCA, respectively. These results prove that non-linear methods have, in fact, a better performance.

A different non-linear DR technique, Kernel PCA (KPCA), is explored in [28]. KPCA technique is similar to PCA, the difference lies in the first step. In KPCA, firstly, a kernel function is applied in order to project the data into a higher dimensional space so that it is subsequently possible to linearly separate data utilising the PCA method. One advantage of this technique over PCA is that KPCA can expose non-linear patterns; however, it requires a lot of computational power, which is a problem for large datasets [29] [30]. Furthermore, obtaining the inverse transformation to decode data is not straightforward.

In [28], three methods were compared: PCA, KPCA and nAEN. This comparison was made on an open dataset introduced by Glauser et al. in [31]. This dataset includes a variety of hand poses performed by 10 participants and stands out for its large size, with more than a million samples. These samples were divided into three smaller datasets for training, validation and testing.

For the nAEN, two architectures were tested, with the one that provided better results consisting of 5 hidden layers. Activation functions in hidden layers were: sigmoid, Tanh, Tanh, ReLU and Scaled Exponential Linear Unit (SELU), respectively, from input to output. The utilised optimiser was Adam, as in [27]. Lastly, the used loss function was Mean Square Error (MSE).

Analysing the results, KPCA showed a better performance than PCA, but the best results were achieved with nAEN, with a reduction of MSE up to 50% in relation to PCA. Additionally, nAEN performs better in realistically reconstructing hand shapes, as seen in Figure 3.5.

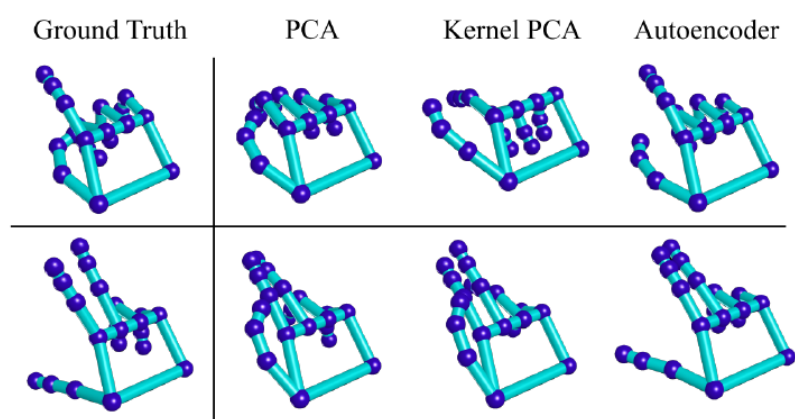


Figure 3.5: Hand pose reconstructions comparison (utilizing 3 PC) [28]

After this analysis, it is possible to conclude that PCA is currently being replaced by new techniques, namely by nAEN [27] [28], achieving this way better results. However, due to its simplicity of application and satisfactory results, PCA is not yet entirely out of use [26].

3.3 Kinematic Retargeting

Transitioning to the topic of the actual robotic hand control, a comprehensive examination of existing methods in the literature was conducted, along with a careful analysis of their outcomes.

Since the Shadow Hand is a humanoid robotic hand, i.e., it has a kinematic configuration identical to the human hand, a first approach might consist of direct retargeting. In this case, the angles obtained from the human hand would be directly mapped to the robotic hand.

Although it has the advantage of being the simplest conversion method and seems to be theoretically accurate, this method has two main problems. First, since the mapping takes place at the level of joint angles, a difference in phalanx length between the human and robotic hand leads to the same angles applied to the robotic hand resulting in a slightly different position. The second problem results from this slight mismatch between the two positions. Since the tasks to be performed by the robotic hand are very precise grasps, it is essential to efficiently replicate the hand's position, especially the position of the fingertips that will perform the grasp, which does not happen with the application of this method.

3.3.1 DexPilot

In [32], a kinematic retargeting optimisation algorithm specifically designed for precision grasps is introduced, explained and tested.

This algorithm prioritises the geometric space of the fingertips. For this purpose, a cost function is defined that considers the distance and relative direction between each fingertip. This cost function considers particular cases that occur in precision grasps, such as ensuring that minimal distances between fingertips in pinching positions are replicated in the robotic hand. Afterwards, the retargeting is performed based on the minimisation of the cost function, thus ensuring that the relative positions between the fingertips in the robotic hand remain as close as possible to the human hand.

In the paper, this algorithm is utilised as a part of a vision-based teleoperation system, DexPilot, for the teleoperation of an Allegro Hand, a robotic hand with four fingers and a high number of DoF, coupled to a KUKA LBR iiwa 7 R800 series robotic arm.

With the application of the explained approach, the authors concluded that their system could be reliably utilised in order to solve several tasks, including precision grasps tasks such as opening and removing paper money from a wallet and stacking a set of three cubes on top of each other. Furthermore, a success rate of 100% was obtained for the majority of the tasks considered in the paper, as can be seen in Figure 3.6. Figure 3.7 shows some retargeting results achieved by the mentioned algorithm.

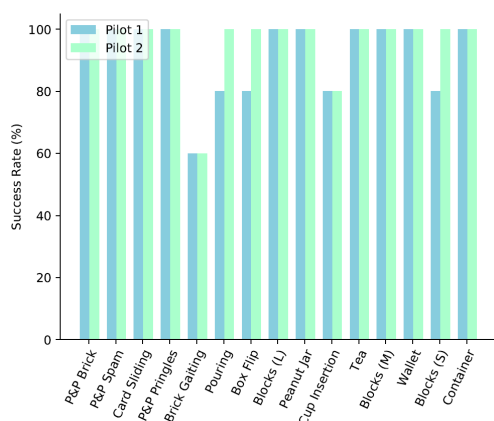


Figure 3.6: Success rate of tasks across two pilots run over five successive trials without re-set [32]

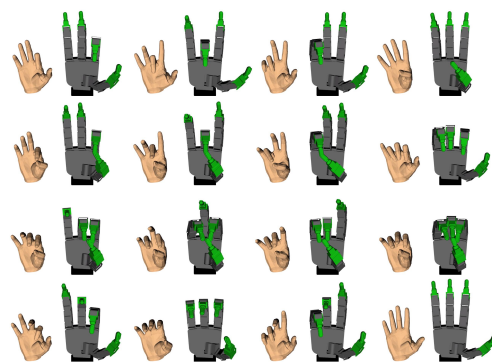


Figure 3.7: Kinematic retargeting results between the human hand model and the Allegro hand model [32]

3.3.2 BioIK

Another algorithm to deal with inverse kinematics is BioIK, first introduced in [33]. Then, a clean and high-performance C++ version of the same algorithm was presented in [34].

BioIK is an open-source software package for ROS that utilises a bio-inspired optimisation algorithm to solve complex optimisation problems, such as inverse kinematics. The key point of BioIK is that the optimisation is done based on a set of weighted goals defined by the user, providing this way high flexibility and control. Furthermore, there are a total of 18 goal types. These goals include the fundamentals, such as position, orientation, and pose (position and orientation) goals, but also include more complex goals, such as:

- Minimal Displacement Goal: tries to keep each joint angle as close as possible to the previous one;
- Centre Joints Goal: tries to keep each joint centred at the respective joint limits;
- Avoid Joint Limits: similar to Centre Joints Goal;
- Look At Goal: tries to align the orientation of a specified link to a goal position;
- Maximum Distance Goal: tries to keep the position of a specific link within a maximum range from a goal position.

In order to exemplify the functionality of BioIK, the authors present some demonstrations, including one with Shadow Hand coupled to the KUKA LWR 4+ robotic arm. In this demonstration, the BioIK was utilised to plan the robotic hand and arm motions for turning a wheel button on an audio mixer, as seen in Figure 3.8. For this task, a tripod grasp was used, being the three fingers (thumb, first and middle fingers) controlled by the algorithm. The final trajectory was achieved by combining a set of 200 solutions from BioIK. With this approach, Shadow Hand was able to successfully turn the wheel button without slipping off.

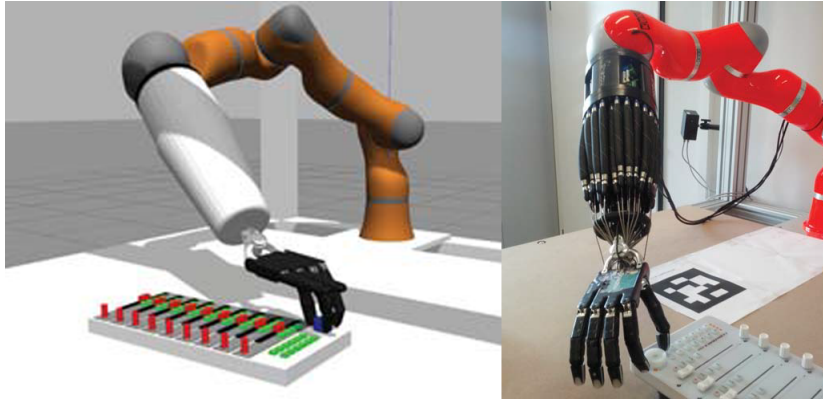


Figure 3.8: Shadow Hand experiment performed in Gazebo simulation and in reality [34]

3.3.3 Hand Embodiment

In [35], Alexander Fabisch et al. proposed a method based on automatic embodiment mapping to transfer human hand motions to robotic systems.

The presented method consists of a modular framework, adaptable to different input modalities by adjusting the record mapping and flexible to other robotic hands by adjusting the embodiment mapping.

Firstly, to record the human hand movement, the authors utilised a set of 13 passive markers coupled to a glove, three of them located at the back of the hand, with the purpose of defining the hand referential, and two passive markers by each finger, one located at the fingertip and the other at the knuckle. To track these markers, the Qualisys MOCAP motion capture system was used. The record mapping is applied to map the human hand into a MANO (hand Model with Articulated and Non-rigid defOrmations) [36] model state. MANO model consists of a realistic 3D hand model that can fit any human hand. Then, an embodiment mapping is performed. This mapping is responsible for translating the MANO hand state to the robotic hand. For this, first, the robotic hand's pose is matched to the MANO hand. Subsequently, each finger position of the robotic hand is individually adjusted to fit the respective finger position in the hand model. Just to note that one of the robotic hands considered in the paper was Shadow Hand.

The approach proposed by the authors was tested in a real system, utilising the MIA hand, a five fingers anthropomorphic robotic hand coupled to the robotic arm UR10e. The test consisted of the grasp and transportation of an insole, initially positioned closer to the hand. After a set of 80 executed grasps, the overall success rate was 83.8%, considering successfully a grasp where the insole didn't fall during the test.

After this analysis, it is possible to conclude that this is an innovative technique with interesting results.

3.3.4 TeachNET

In [37], Shuang Li et al. proposed a new approach to solve the kinematic retargeting problem based on neural networks.

In this paper, the authors presented TeachNet, a novel end-to-end neural network architecture that, from depth images of the human hand, estimates the joint angles to be applied in the robotic hand, namely the Shadow Hand, to reproduce the human hand shape.

The task of directly solving joint regression problems from human hand images is known to be a particularly challenging one, mainly due to the fact that the robot hand and the human hand occupy two distinct domains. Considering this, TeachNet consists of two branches, the robot hand branch that acts as the teacher and the human hand branch as the student. Each branch has as input depth images and as output joint angles. Between these two branches, there is a consistency loss with the purpose of aligning both features. To better understand the network architecture, it is presented in Figure 3.9.

In order to train the developed network, the authors created a new dataset utilising the existing dataset BigHand2.2M [38] and the BioIK solver mentioned earlier. Depth images were extracted from the BigHand2.2M, and the respective hand keypoints were mapped into Shadow Hand joint angles by BioIK. The solver was set with the following goals: mapping of the fingertip positions with a weight of 1, mapping of the proximal interphalangeal joint positions with a weight of 0.2, and lastly, mapping proximal and distal phalanges with a weight of 0.2.

Finally, to evaluate the proposed method, several hand gestures were performed by five novice teleoperators and replicated by Shadow Hand. This test resulted in a success rate of 78.26%, being the successful poses presented in Figure 3.10. Furthermore, manipulation experiments were also performed and compared regarding the necessary time to accomplish a specific task with the DeepPrior++ [39] method. The results show that by utilising TeachNet, the average time for each task execution was only 57% compared to the time needed by utilising DeepPrior++.

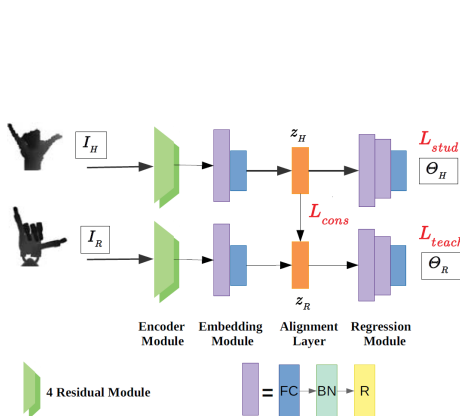


Figure 3.9: TeachNet architecture [37]



Figure 3.10: Successful teleoperation results [37]

Considering the good results achieved by each of the aforementioned methodologies, they will be adjusted, if required, to be compatible with the Shadow Hand and subsequently applied

and compared in order to infer which one best adapts to the tasks to be performed by the hand.

Taking into account this literature review, a noteworthy observation was drawn, indicating that the utilisation of synergies is not essential. This observation is based on the aforementioned algorithms and in the Shadow Hand, which is already prepared for high-level control. Given that the presented algorithms directly map the position of the human hand to the corresponding joint angles of the robotic hand, the incorporation of synergies was deemed unnecessary. Nonetheless, the study conducted on data acquisition, considering the concept of synergies, can be effectively utilised as input for the presented algorithms, thus, it will not be disregarded.

3.4 Learning from Demonstration

Learning from Demonstration (LfD), as known as Learning by Demonstration (LbD), is a powerful technique involving training a robot system to perform a task by demonstrating it instead of explicitly programming it. LfD has the potential to speed up the process of teaching robots new skills, and it can be especially beneficial for tasks that require a high level of dexterity and precision. Next, some papers about this topic will be analysed.

3.4.1 Gaussian Mixture Regression

In [40], LfD is applied in order to teach a robotic system how to grasp and manipulate an object, also taking into account the grasping force. The robotic system comprised a three-fingers hand (BarrettHand¹) and a 6 DoF arm. This paper utilised an interesting method to measure the fingertip force on the object. It was done by observing the colouration changes in the nail and surrounding skin with a camera. The data (motion and force) was acquired by two cameras on opposite sides of the object in order to avoid occlusions.

The demonstration model was then created by encoding the captured data with a Gaussian Mixture Model (GMM). GMM is a probabilistic model that calculates a probability distribution of the data. Finally, a smooth movement trajectory (position and orientation) and force are given by the created model using Gaussian Mixture Regression (GMR). The force was directly applied to the hand, and the motion was given through inverse kinematics applied to the robotic set, considering the desired position and orientation of the end effector.

Applying the described method, Yun Lin *et al.* [40] were able to reproduce the pick-and-place of several objects properly, controlling not only the motion but also the applied force.

3.4.2 Dynamic Movement Primitives

In article [41], a different approach is followed instead of utilising GMM. In this case, the robotic set consisted of a 7 DoF arm and a parallel gripper. The adopted procedures consisted of three main steps: hand movements track, hand movements segmentation and finally, application of the Dynamic Movement Primitives (DMPs) framework.

¹<https://advanced.barrett.com/barretthand>

The first step was performed by utilising a Kinect camera to track human hand demonstration movement. To ease hand tracking, a colour-marker glove was utilised. From this step, three essential outputs were obtained: hand position, hand orientation and distance between fingers.

The second step was to divide all hand movements into fractions of simple movements, termed motion primitives. For example, the task of dispensing water from a water thermos pot was segmented into a sequence of four actions: pick up the cup, place the cup in the right place, press the dispenser button and lastly, pick up the cup.

In the third step, each motion primitive was encoded by DMPs. DMPs are a framework for learning and generating smooth and adaptable movements in robotics and control systems. The key point here is the adaptability provided. For instance, with DMPs, if the target object to pick up change its position, the robot is able to adapt to this new position. This feature is essential in dynamic environments, such as industries.

Finally, the inputs to control the entire robot trajectory are reproduced by the sequence of DMPs models of the motion primitives. As in [40], the trajectory was obtained through inverse kinematics applied to the end effector position and orientation and the parallel gripper control was done by directly applying the distance between fingers.

Utilising the described procedure, the authors were able to achieve the desired results. The robot set was able to replicate the given instruction and even modify its trajectory and orientation in order to adapt to environmental changes.

Another example of the utilisation of DMPs is in [42]. In this paper, the authors combined DMPs, Virtual Springs Framework (VSF) and force feedback to effectively achieve manipulation of unknown objects. The utilised hardware consisted of a four fingers Allegro Hand² robot with 16 DoF and a 6 DoF UR5 arm. The objective was to teach the robot with just one demonstration, and then the robot should be able to reproduce the demonstration, even with position changes or different object sizes.

Demonstrations were recorded kinesthetically, i.e., the human expert physically held and guided the robot to the desired position with the desired trajectory. This method is interesting because the data is directly acquired from the robot, so no further conversions and adaptations are needed. However, it limits hand dexterity because moving all the fingers simultaneously is challenging.

Subsequently, the control technique was divided into two levels. DMPs were utilized to learn the object's motion, and VSF was utilised to control the fingertip forces to keep the object in grasp. Briefly explaining, VSF consists in defining virtual springs between the object and the fingers, as shown in Figure 3.11. Then, the grasp force is calculated, considering the forces generated by the virtual springs. Moreover, the force feedback at the fingertips can be used to adjust the virtual springs' stiffness.

The experiments were done with three different size rectangular prismatic objects and different initial and final states in order to test the robustness of the implemented algorithms.

²<https://www.wonikrobotics.com/research-robot-hand>

The results showed that an adaptive stiffness, given by force feedback, improved the grasp stability and increased the final pose accuracy. In addition, results also showed that it was possible to generalise a learned hand manipulation to new conditions, with a small average final error of position and orientation. Specifically, errors of less than a centimetre and around 16 degrees were obtained, for position and orientation, respectively.

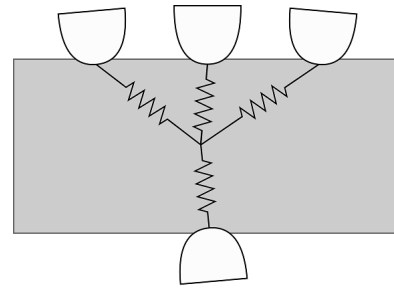


Figure 3.11: Virtual Spring Framework represented in an object grasped by four fingers

3.4.3 Concept Constrained Learning from Demonstration

In [43], a new algorithm for robust LfD is introduced, the Concept Constrained LfD (CC-LfD). This method was utilised to control a parallel gripper coupled to a 7 DoF robotic arm (Rethink Robotics Sawyer³). The tests made consisted of kinesthetically training the robot with low-quality demonstrations and then skill repairing the respective task demonstration using a single constraint-annotated demonstration (CC-LfD technique) or by adding high-quality demonstrations to the training procedure.

The CC-LfD method consists of, as said, adding a constraint-annotated demonstration. The application of these constraints is made over time by annotating a frame. These constraints are boolean expressions that describe the correct behaviour to adopt. To give a practical example, when transporting a full liquid cup, one constraint could be keeping the cup upright to not spill the liquid. Theoretically, with this method, poor demonstrations⁴ could be corrected rapidly.

In order to test and compare the CC-LfD method, two tasks were tested. The first one involved picking up a cup, transporting it over another cup and transferring its content into it without spilling along the way. The second task consisted in picking up a cup, moving it around an obstacle and resting the cup on top of the obstacle.

For each task, a total of 3 poor demonstrations were given during the training process. Then, with the goal of correcting the learned skills, a single constraint-annotated demonstration was added up. The results were compared to the addition of up to 28 high-quality demonstrations (repairs) to the initial three poor demonstrations.

Analysing Table 3.1, the CC-LfD technique resulted in a success rate of 90% and 100% for the first and second tasks, respectively. On the other hand, with the 28 repairs, the first task achieved a success rate of 60% and the second task 70%. Comparing these results, it is possible to conclude that CC-LfD is very effective by dramatically repairing poor skills.

³<https://www.rethinkrobotics.com/sawyer>

⁴poor demonstration: demonstration where the constraints are violated

Table 3.1: First and second tasks performance results [43]

Type	Poor Demonstrations	Repairs	Attempts	Correct (1st 2nd task)	Incorrect (1st 2nd task)
Baseline	3	0	10	0 1	10 9
7 Repairs	3	7	10	4 5	6 5
14 Repairs	3	14	10	6 4	4 6
21 Repairs	3	21	10	7 7	3 3
28 Repairs	3	28	10	6 7	4 3
Constrained	3	1	10	10 9	0 1

3.4.4 Artificial Neural Networks

In [44], a different LfD approach is followed based on artificial neural networks. The goal was to learn manipulation skills from human demonstration and then transfer them to real robots.

The data acquisition phase was done by an instrumented glove and by cameras. The glove captured finger motion and tactile information, and cameras recorded human gestures resorting to red Light-Emitting Diodes (LEDs) placed on the fingertips and hand joints.

The data obtained were then utilised to train neural networks. In order to better adapt to the task complexity, a few neural networks were tested.

The first task to be performed was a pick-and-place task. The goal of this task was for the robot to grasp a box and place it onto a plate. Both objects were equipped with tracking markers (LEDs). Ten human demonstrations were performed for training with both items in different positions and orientations. According to the simplicity of this task, a simple feed-forward network was utilised. As a result, the robot could successfully complete the pick-and-place task, even with the items placed in different places from demonstrations.

Another task consisted in opening a beverage bottle. This task can be divided into three steps: turn the lid until it has been loosened, grasp it, and place it next to the bottle. Applying the feed-forward network to this task resulted in an infinite number of lid rotations. This happened because the robot was unable to determine the time at which the lid could be removed since the utilised network had no memory and used no tactile information. In order to overcome this task, a recurrent network was used. The recurrent connections in the model of this network enable it to remember past actions. The application of these networks led to the successful performance of the robot in opening a beverage bottle.

In article [45], the OpenAI⁵ team used Reinforcement Learning (RL) to learn manipulation techniques in order to reorient a cube in a specific orientation. The hardware used was a Dexterous Shadow Hand (the same robotic hand to be utilised in dissertation work).

A technique different from all those already mentioned was utilised for data acquisition. It was done 100% by simulation in Multi-Joint dynamics with Contact (MuJoCo) [46]. With the goal of better approximating the simulation to the real-world scenario, some randomisation was applied, such as observation noise, physics parameters and even visual appearance by changing,

⁵<https://openai.com/>

for instance, the background colour. From the simulation, two classes of data were extracted. Robot fingertips positions and object position and orientation were acquired utilising vision-based methods.

The fingertip positions were given as input to train a manipulation control policy using RL. The cube orientation was predicted using a convolutional neural network with images of the object from three different orientations as input. Finally, the pose estimation network and the control policy were combined in order to transfer the learned manipulation methods to the real robot.

The results obtained were very satisfactory, being the robotic hand capable of successively reorienting the cube in the specified orientation, as can be seen in <https://youtu.be/DKe8Fum0D4E>. From no real human manipulation data, the RL algorithm proved to be able to perform the proposed task, demonstrating its power for this type of application.

Reaching the end of this section, it is possible to realise that quite a few LfD methods already exist, including methods based on artificial neural networks. As the project progresses, it will be chosen the one that best fits the scope of the dissertation.

3.5 Automotive Painting Quality Control

Automotive paint quality inspection evaluates the appearance and finish of a vehicle's paint job. This includes checking for defects such as inclusions, cracks, oversprays, etc. Paint quality control is an essential step in the manufacturing process as it helps to ensure that vehicles are delivered with a high-quality finish which is visually appealing to customers. This section will give an overview of state-of-the-art quality inspection technologies.

An emerging quality control technology, especially in automotive and aviation industries, is the utilisation of terahertz waves [47]. This method provides high-speed volume inspections, as well as high-resolution thickness determination, which is interesting in the scope of painting works. Another great advantage is that this is a non-destructive method since it is contactless performed. This means the quality inspection utilising terahertz waves will not affect the paint properties.

In the particular case of paint quality inspection, this technology presents a significant advantage. With terahertz waves, it is possible to measure the thickness of the paint when it is still wet, unlike other sensors that would leave their mark and damage the paint. Since it is possible to measure paint quality during the painting process, if the paint needs to be added for thickness correction, this process is performed over the wet paint, leading to better quality results.

One of the disadvantages of this method is the cost of terahertz waves systems; however, it tends to reduce as the technology utilised in these sensors evolves.

In article [48], a different approach was presented and followed. Mao *et al.* tested the utilisation of a deep learning network applied to captured rims images to inspect their quality. Although this is not exactly the study case, the same methodology can be applied.

For training the network, a total of 245 images of defects were collected, containing a total of 270 defects. Since the use of neural networks requires a high number of training samples in

order to achieve reasonable accuracy, an expansion of the dataset was performed. To this end, a Deep Convolutional Generative Adversarial Network (DCGAN) was utilised.

Afterwards, a YOLO v3 and a YOLO v4 network were trained to recognise the defects present in the captured images or, eventually, their absence. These defects included, for example, dirt spots and paint stains.

Once the networks were trained, the best result was achieved with the YOLO v4 in conjunction with DCGAN, resulting in an accuracy of 86.8%.

After reviewing the aforementioned techniques for quality inspection, the utilisation of deep learning networks seems to be the solution that best fits the dissertation application case. It has the advantage of a low-cost implementation combined with good accuracy.

Chapter 4

Implementation

This chapter will demonstrate in detail the work undertaken within the scope of this dissertation. It delves into subjects and methodologies previously outlined in Chapter 3, namely data acquisition and kinematic retargeting algorithms. All the code mentioned in this chapter is publicly available at https://github.com/BrunoSantosCode/ShadowHand_MastersDissertation.

4.1 Data Acquisition

Regarding data acquisition, an introductory study was conducted on the utilisation of neural networks for acquiring keypoints of the human hand from two-dimensional coloured images [49]. Nevertheless, based on the achieved results, alternative methodologies were tested.

An additional approach went through testing the sensorial glove [20] referred to in Subsection 3.1.2. The utilisation of this glove consisted of an initial calibration of the sensors at a known hand position, and, from this point on, the IMUs sensors were responsible for tracking the changes in the hand position. In order to obtain the position, the acceleration values provided by the sensors were integrated twice. Despite working with some accuracy in the first instants of time, due mainly to the errors obtained in the successive integrations, the results quickly deviated from the hand's real position, making the glove an unfeasible solution.

Therefore, another solution was tested based on vision techniques. The chosen method was OpenPose [50]. OpenPose is a method based on a convolutional neural network and was the first real-time, multi-person system to detect collectively human body, hand, facial and foot keypoints, totalling 135 keypoints. One of the reasons for the choice of this method is its high accuracy. OpenPose is capable of estimating the referred keypoints even in challenging environments, such as when occlusions occur or in cluttered backgrounds. Its accuracy has been proven on two datasets, namely MPII [51] and COCO [52]. In both datasets, OpenPose outperformed the previous methods, achieving this way state-of-the-art results. Another advantage of this method relies on its real-time performance, which is essential for many application cases, such as robotic hand teleoperation. Furthermore, OpenPose is an open-source method, meaning it is freely available

for research utilisation. Lastly, its effectiveness is proven by the high number of citations, already exceeding fifteen thousand.

For the specific application case of the dissertation, the combination of OpenPose and a Microsoft Kinect v2 was employed to detect 3D keypoints of the human hand. The Kinect provided 2D images, which were processed by OpenPose to detect the corresponding 2D points. Subsequently, the depth information was extracted from the depth sensor of the Kinect.

4.2 Teleoperation

Having established a method for capturing the position of the human hand, the subsequent step involved implementing teleoperation for the Shadow Hand to comprehend its practical dexterity. This step considers the kinematic retargeting methods mentioned in Section 3.3.

The entire code base was developed within the Robot Operating System (ROS) environment using Python programming language. The development process occurred within the dedicated docker container provided by the Shadow Hand company, encompassing all essential software components required for high-level hand control.

4.2.1 Direct Joints Mapping

In the initial phase, the primary focus was to establish integration between OpenPose and the Shadow Hand. To achieve this, a direct mapping of joint angles was implemented.

To this end, the joint angles were extracted from the acquired keypoints, depicted in Figure 4.1, employing the vector dot product expression:

$$\vec{a} \cdot \vec{b} = |\vec{a}| |\vec{b}| \cos \theta \quad (4.1)$$

Solving relatively to θ :

$$\theta = \cos^{-1} \left(\frac{\vec{a} \cdot \vec{b}}{|\vec{a}| |\vec{b}|} \right) \quad (4.2)$$

Due to the straightforward nature of the followed approach, only a subset of joints was considered in this phase. Specifically, three joints per finger were taken into account, resulting in a total of 15 joints.

Thumb:

- Joint 1 angle defined with $\vec{a} = \overrightarrow{KP_2 KP_3}$ and $\vec{b} = \overrightarrow{KP_3 KP_4}$
- Joint 3 angle defined with $\vec{a} = \overrightarrow{KP_1 KP_2}$ and $\vec{b} = \overrightarrow{KP_2 KP_3}$

¹https://cmu-perceptual-computing-lab.github.io/openpose/web/html/doc/md_doc_02_output.html

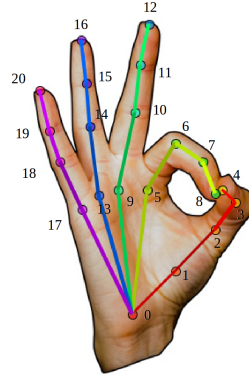


Figure 4.1: OpenPose hand output format (from Openpose documentation¹)

- Joint 4 angle defined with $\vec{a} = \overrightarrow{KP_0 KP_1}$ and $\vec{b} = \overrightarrow{KP_1 KP_2}$

Index finger:

- Joint 1 angle defined with $\vec{a} = \overrightarrow{KP_6 KP_7}$ and $\vec{b} = \overrightarrow{KP_7 KP_8}$
- Joint 2 angle defined with $\vec{a} = \overrightarrow{KP_5 KP_6}$ and $\vec{b} = \overrightarrow{KP_6 KP_7}$
- Joint 3 angle defined with $\vec{a} = \overrightarrow{KP_0 KP_9}$ and $\vec{b} = \overrightarrow{KP_5 KP_6}$

Note that KP_x corresponds to the x keypoint observed in Figure 4.1 and consequently $\overrightarrow{KP_x KP_y}$ corresponds to the vector starting at x and ending at y .

The remaining fingers were defined similarly to the index finger. It is noteworthy that, for the calculation of the knuckle angle (Joint 3), the $\vec{a}$ was purposely selected as $\overrightarrow{KP_0 KP_9}$ in all fingers, except the thumb, as a way to try to obtain the angle between the proximal phalanges and the palm.

The remaining Shadow Hand joints that were not calculated were set to zero degrees.

The final step involved assigning the calculated joint angles to the robotic hand, which was done by calling the method “move_to_joint_value_target_unsafe” method from the SrHandCommander, a high-level interface to control the robotic hand.

During testing, it was observed that specific keypoints were frequently detected incorrectly by OpenPose, leading to significant deviations from their actual positions. To address this issue, a median filter was applied to the acquired keypoints position. This filter was chosen due to the low influence of outliers on the filtered result. Figure 4.2 illustrates the effectiveness of the median filter in removing outliers, such as the wrongly detected index fingertip by OpenPose.

To ensure the method’s flexibility to various forms of human hand shape acquisition, a dedicated ROS message was created for hand keypoints. This message consists of an array of geometry_msgs/Point, a specific ROS message for storing the position of a point in free space.

Summarizing, the end-to-end integration went through the following steps, as shown in the Unified Modeling Language (UML) sequence diagram presented in Figure 4.3:

1. OpenPose publishes a ROS message with the detected keypoints;

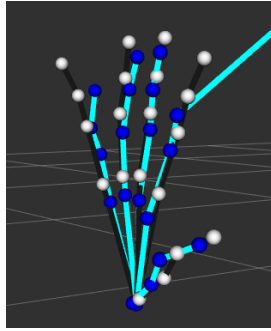


Figure 4.2: Hand keypoints before (in blue) and after (in white) the application of the median filter

2. A conversion is performed between the OpenPose ROS message and the standard hand keypoints message, the latter being published in a ROS topic;
3. The ROS topic containing the keypoints message is subscribed, and in the respective callback function², the keypoints are saved into an array with previous keypoints;
4. A thread uses the filtered keypoints, fetching the median of each of the positions of the vector mentioned in the previous point;
5. The same thread performs the respective transformation, from the filtered keypoints positions into angles taking into account the mentioned approach;
6. Finally, and also in the same thread, the angles are sent to Shadow Hand through the SrHandCommander interface.

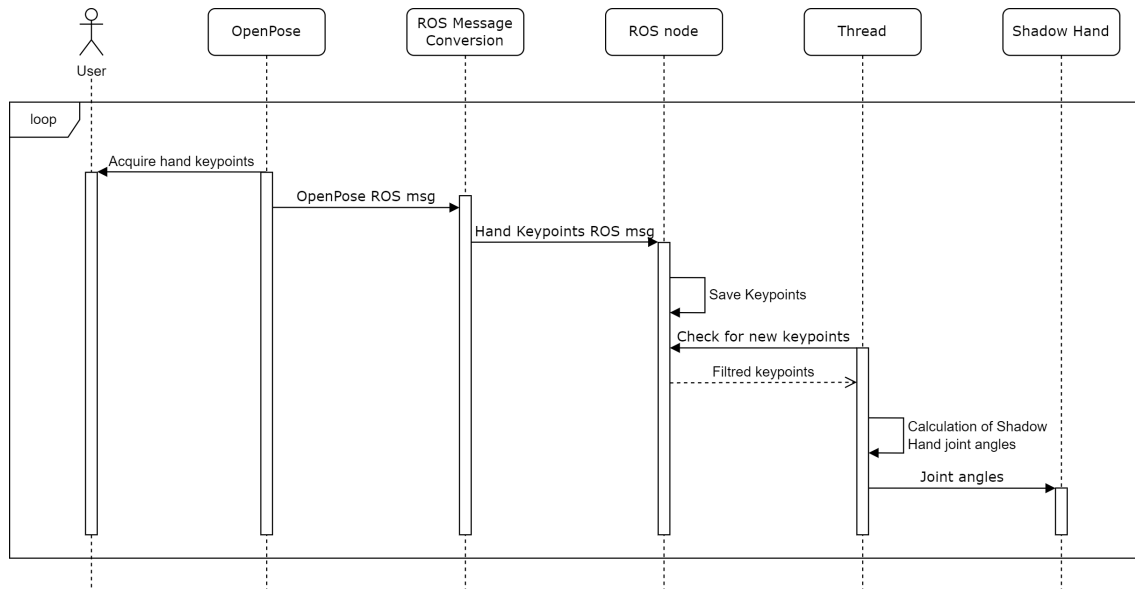


Figure 4.3: Program architecture represented in a UML sequence diagram

The aforementioned structure was followed for the remaining kinematic remapping approaches.

²callback function: a ROS function that is called every time a new message arrives to the subscribed topic

4.2.2 DexPilot

Having accomplished the end-to-end integration, the subsequent phase consisted of implementing and testing the kinematic retargeting algorithms presented in Section 3.3, starting with DexPilot.

This algorithm prioritises the geometric space of the fingers, being the algorithm built on vectors that connect the fingertips of the hand, as seen in Figure 4.4. Since the robotic hand to be used has five fingers, extra vectors were added in the implementation of this algorithm. In addition, for reasons explained further on, vectors from the fingertips to the wrist were also added. The resulting number of vectors totalled 14, as shown in Figure 4.5.

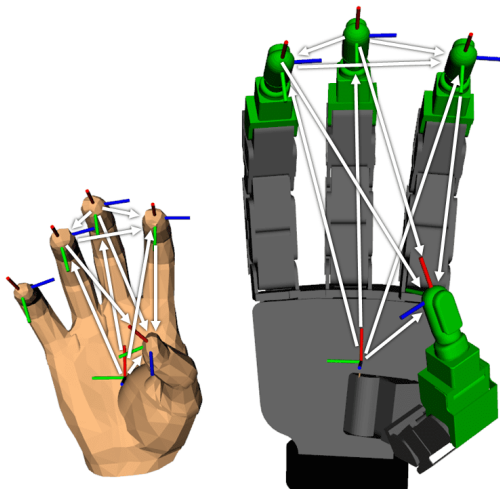


Figure 4.4: Space vectors utilised in DexPilot for both the human hand model and Allegro hand [32]

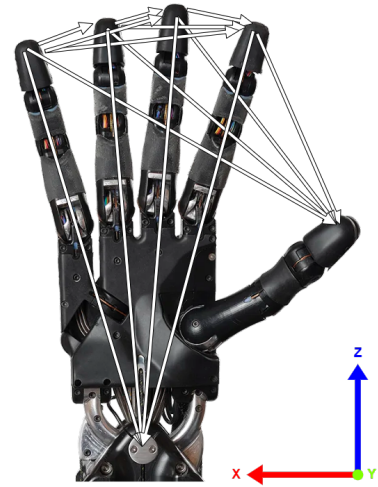


Figure 4.5: Space vectors adapted for Shadow Hand

The vectors defined in [32] were divided into two sets: S_1 that included the vectors from primary fingers (index, middle and ring) and pointed to the thumb; S_2 that included the vectors between two primary fingers. Furthermore, in the adaptation of the algorithm, the set S_3 was also added, as mentioned, including the vectors from the fingertips to the wrist.

The optimisation algorithm cost function utilised in DexPilot was based on these vectors, and it is given by:

$$C(q_h, q_r) = \frac{1}{2} \sum_{i=0}^N s(d_i) \|r_i(q_r) - f(d_i) \hat{r}_i(q_h)\|^2 + \gamma \|q_r\|^2 \quad (4.3)$$

In 4.3, q_h and q_r correspond to the angles of the human hand and Allegro hand, respectively, and $r_i(q_r)$ corresponds to the vectors of the human hand. In addition, d_i are the length of the human hand vectors:

$$d_i = \|r_i(q_h)\| \quad (4.4)$$

and $\hat{r}_i(q_h)$ are the corresponding unit vectors:

$$\hat{r}_i(q_h) = \frac{r_i(q_h)}{\|r_i(q_h)\|} \quad (4.5)$$

The $s(d_i)$ is the switching weight function defined by:

$$s(d_i) = \begin{cases} 1, & d_i > \varepsilon \\ 400, & d_i \leq \varepsilon \quad \wedge \quad r_i(q_h) \in S_2 \\ 200, & d_i \leq \varepsilon \quad \wedge \quad r_i(q_h) \in S_1 \end{cases} \quad (4.6)$$

where ε corresponds to a proximity threshold; in his way, vectors with short lengths associated with precision grasps (e.g., pinching) were given higher weights so that they would have more priority.

The $f(d_i)$ is the distancing function defined as:

$$f(d_i) = \begin{cases} \beta d_i, & d_i > \varepsilon \\ \eta_1, & d_i \leq \varepsilon \quad \wedge \quad r_i(q_h) \in S_1 \\ \eta_2, & d_i \leq \varepsilon \quad \wedge \quad r_i(q_h) \in S_2 \end{cases} \quad (4.7)$$

where β is a scaling factor between the human and the robotic hand, defined as 1.6; η_1 , defined as 1×10^{-4} m, is responsible for closing the distance between a primary finger and the thumb if the distance between these fingers in the human hand is inferior to the ε threshold; η_2 defined as 3×10^{-2} m forces a minimum separation distance between two primary fingers, once again, if the distance between these fingers in the human hand is inferior to the ε threshold.

Lastly, the γ , defined as 2.5×10^{-3} , is a weight on regularising the robotic hand angles to zero (open hand position). The authors found this term helpful for reducing redundancy in the solution.

For the algorithm implementation, the human hand fingertips vectors ($r_i(q_h)$) were directly acquired from the fingertips positions given by OpenPose. After calculating these vectors, a referential frame transformation was performed to the frame shown in Figure 4.5 by a rotation matrix since the relevant parameters of the vectors for the algorithm application were only direction and length. Considering the new referential frame as 1 and the original as 0, the rotation matrix was defined as:

$$R_0^1 = (R_1^0)^T = \begin{bmatrix} x_1^0 & y_1^0 & z_1^0 \end{bmatrix}^T, \quad (4.8)$$

where:

$$\begin{aligned}
z_1^0 &= \frac{\vec{v}_{09} + v_{013}}{2} \\
z_1^0 &= \frac{z_1^0}{\|z_1^0\|} \\
y_1^0 &= v_{09} \times v_{013} \\
y_1^0 &= \frac{y_1^0}{\|y_1^0\|} \\
x_1^0 &= y_1^0 \times z_1^0
\end{aligned} \tag{4.9}$$

In 4.9, v_{xy} corresponds to the vector from the keypoint x to the keypoint y, being x and y the keypoint number, as illustrated in Figure 4.1.

After having the rotation matrix, the points in the new referential frame were obtained using the following formula:

$$\vec{v}^1 = R_0^1 \cdot \vec{v}^0 \tag{4.10}$$

The Shadow Hand fingertip positions were calculated utilising direct kinematics from the angles of the hand joints. The direct kinematics were calculated by the Denavit-Hartenberg (DH) parameters by the analysis of the hand kinematic model, presented in Chapter 2, in Figure 2.4.

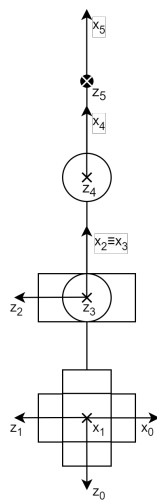


Figure 4.6: Thumb kinematic model

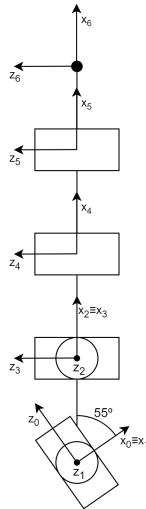


Figure 4.7: Little finger kinematic model

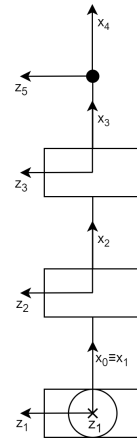


Figure 4.8: Remaining fingers kinematic model

Table 4.1: Thumb DH parameters

i	a_i (cm)	α_i (degrees)	d_i (cm)	θ_i (degrees)
1	0	90	0	θ_1^*
2	4.5	0	0	θ_2^*
3	2.50	0	0	θ_3^*
4	2.60	0	0	θ_4^*

Table 4.2: Little finger DH parameters

i	a_i (cm)	α_i (degrees)	d_i (cm)	θ_i (degrees)
1	0	90	0	θ_1^*
2	6.589	0	0	55
3	0	-90	0	θ_2^*
4	4.5	0	0	θ_3^*
5	2.5	0	0	θ_4^*
6	2.6	0	0	θ_5^*

Table 4.3: Remaining fingers DH parameters

i	a_i (cm)	α_i (degrees)	d_i (cm)	θ_i (degrees)
1	0	90	0	θ_1^*
2	4.50	0	0	θ_2^*
3	2.50	0	0	θ_3^*
4	2.60	0	0	θ_4^*

From the DH parameters, the transformation matrices were defined as:

$$T_i = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cdot \cos \alpha_i & \sin \theta_i \cdot \sin \alpha_i & a_i \cdot \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cdot \cos \alpha_i & -\cos \theta_i \cdot \sin \alpha_i & a_i \cdot \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.11)$$

and the final transformation matrices to calculate the fingertip positions given the joint angles were calculated by successively multiplying the individual transformation matrices.

After having the fingertips' positions, the vectors seen in Figure 4.5 were easily calculated by subtracting the corresponding keypoints' positions.

Lastly, the optimisation algorithm was implemented utilising the *minimise* method from the *scipy* optimiser library in Python. The *minimise* function receives as input the cost function, defined in 4.3, the initial guess of robot joints, defined as the previously calculated joints, the optimisation method, defined empirically as Sequential Least Squares Programming (SLSQP) and finally the joints bounds. In order to deal with the couple of Joint 1 and Joint 2 (explained in Section 2.3), these two joints were replaced by a Joint 0.

After performing some tests on the algorithm, some modifications were done to improve it. Namely, a new set of vectors was added to deal with positions where the fingers closed; for this, a total of 4 vectors pointing from each fingertip of primary fingers to the wrist were added. In addition, some weights were adjusted, and the ε threshold was divided into three thresholds for the three sets of fingers, namely ε_1 , ε_2 and ε_3 for S_1 , S_2 and S_3 , respectively.

The final parameters, weight function and distancing function, were defined as in 4.12, 4.13 and 4.14, respectively.

$$\begin{aligned}
 \varepsilon_1 &= 0.03 \text{ m} \\
 \varepsilon_2 &= 0.03 \text{ m} \\
 \varepsilon_3 &= 0.05 \text{ m} \\
 \beta &= 1.0 \\
 \eta_1 &= 0.0001 \text{ m} \\
 \eta_2 &= 0.03 \text{ m} \\
 \eta_3 &= 0.01 \text{ m} \\
 \gamma &= 0.003
 \end{aligned} \tag{4.12}$$

$$s(d_i) = \begin{cases} 1, & d_i > \varepsilon_1 \wedge r_i(q_h) \in S_1 \\ 1, & d_i > \varepsilon_2 \wedge r_i(q_h) \in S_2 \\ 1, & d_i > \varepsilon_3 \wedge r_i(q_h) \in S_3 \\ 200, & d_i \leq \varepsilon_1 \wedge r_i(q_h) \in S_1 \\ 400, & d_i \leq \varepsilon_2 \wedge r_i(q_h) \in S_2 \\ 100, & d_i \leq \varepsilon_3 \wedge r_i(q_h) \in S_3 \end{cases} \tag{4.13}$$

$$f(d_i) = \begin{cases} \beta d_i, & d_i > \varepsilon_1 \quad \wedge \quad r_i(q_h) \in S_1 \\ \beta d_i, & d_i > \varepsilon_2 \quad \wedge \quad r_i(q_h) \in S_2 \\ \beta d_i, & d_i > \varepsilon_3 \quad \wedge \quad r_i(q_h) \in S_3 \\ \eta_1, & d_i \leq \varepsilon_1 \quad \wedge \quad r_i(q_h) \in S_1 \\ \eta_2, & d_i \leq \varepsilon_2 \quad \wedge \quad r_i(q_h) \in S_2 \\ \eta_3, & d_i \leq \varepsilon_3 \quad \wedge \quad r_i(q_h) \in S_3 \end{cases} \quad (4.14)$$

To conclude, refer that, as the optimisation part takes a considerable amount of time, another thread was created for the optimisation part, and the resultant program architecture is represented in the UML sequence diagram presented in Figure 4.9. The communication between threads was done utilising global variables, namely for hand keypoints positions and for joint angles, being the access protected with mutexes to avoid conflicts.

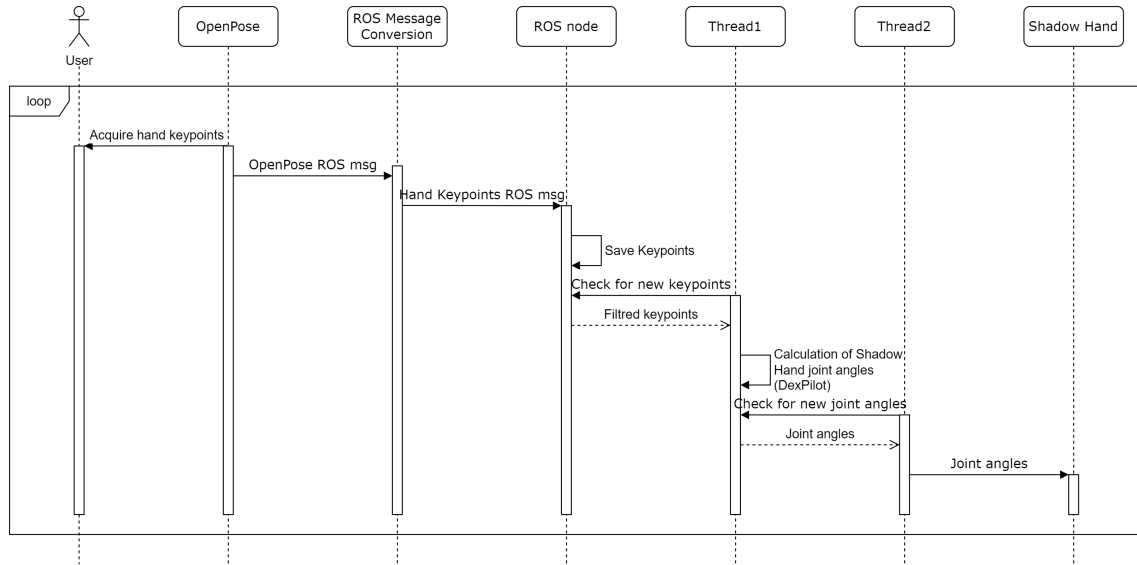


Figure 4.9: Program architecture represented in a UML sequence diagram

4.2.3 BioIK

Subsequently, BioIK was implemented. Since BioIK is exclusively available in C++, the structure of the utilised Python code in the previous algorithms was replicated in C++, preserving the same features as much as possible.

Prior to running the solver itself, some processing was applied to the keypoints in addition to the median filter. Namely, the acquired human hand keypoints were transformed in a way that the links (distance between two keypoints, equivalent to bones distance) had the same size as the respective ones in the Shadow Hand. For this, an iterative process was performed, starting from the wrist and ending at each fingertip. This process consisted of maintaining the direction of each link

and changing its size according to the respective link in the robotic hand. The final objective was to get better results applying this technique since the BioIK solver is based on inverse kinematics. The result of the application of this technique is shown in Figure 4.10. Being Shadow Hand designed to be the same size as a human hand, the difference between keypoints is not substantial, except for the little finger since all the robotic hand primary fingers have the same size, but in reality, the little finger is smaller than the others primary fingers.

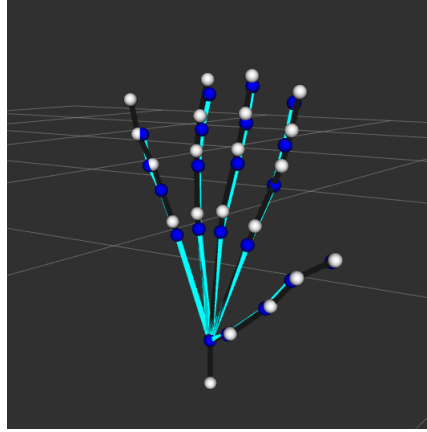


Figure 4.10: Keypoints before (in blue) and after (in white) the application of robotic hand mapping

For the implementation of the BioIK, four types of goal functions were utilised: position, direction, joint function and centre joints goal. First, the fingertip positions were mapped utilising the position goal with weights defined as one since the position of the fingertips is of extreme relevance to precision grasps. The knuckles were mapped also utilising position goals, but this time with inferior weights, namely 0.3. The proximal phalanges direction of each finger was mapped with a weight of 0.1, except for the thumb, where the mapped phalanges were intermediate phalanx and distal phalanx, also with weights of 0.1. In addition, the coupling of Joints 0 and 1 was given to the solver algorithm by a joint function goal with a weight of 1. Lastly, the centre joints goal was utilised with a weight of 0.15 to avoid work on the joints limits.

4.2.4 Hand Embodiment

Regarding the Hand Embodiment algorithm, in [35], the authors have made their implementation publicly available in a GitHub repository³. In addition, some examples of human hand markers position recordings were provided.

The first step here consisted of implementing the provided pipeline with the same structure of already mentioned algorithms and testing the algorithm with available examples. However, the implementation of this algorithm proved to be quite complex due to the multiple configuration files, namely the configuration file to convert the hand markers to the MANO model and, subsequently, the configuration file to pass from the MANO model to the robotic hand. The complete set

³https://github.com/dfki-ric/hand_embodiment

of transformations can be seen in Figure 4.11. Furthermore, after running the implemented program, the pipeline provided by the authors proved to be very slow, taking about 900 milliseconds between receiving the human hand markers and the output of the Shadow Hand joints.

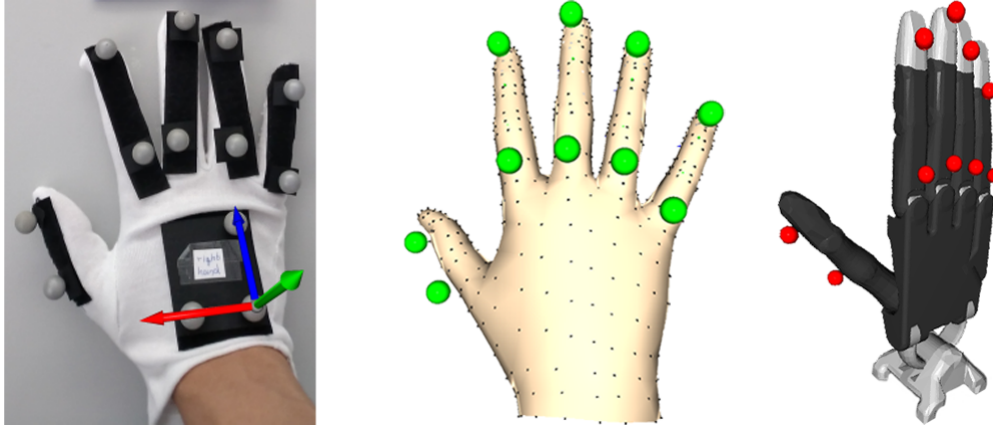


Figure 4.11: Hand Embodiment algorithm steps: human hand markers (left), MANO model (middle) and Shadow Hand mapping (right) [35]

Still, an attempt was made to adapt the algorithm to receive OpenPose keypoints as input since the original algorithm received a total of 13 keypoints. Of these 13 keypoints, as seen in the left part of Figure 4.11, three were located on the back of the hand and were responsible for establishing the hand's referential frame and the remaining ten were located two on each finger, on the fingertip and the knuckles. Considering this, the keypoint of the referential frame origin, in the left part of Figure 4.11, was replaced by the keypoint of the wrist. The remaining palm keypoints were obtained considering the referential frame defined in Figure 4.11 and being at a distance of two centimetres above (positive z-axis) and to the left (negative x-axis) of the wrist. The remaining ten keypoints were directly obtained from OpenPose.

During the testing phase of this adaptation, it became evident that the algorithm did not function as intended. Despite the considerable effort invested in implementing the Hand Embodiment method and attempting to adapt it to OpenPose, the desired results were not achieved. Considering the inherent complexity of the Hand Embodiment approach and the time already dedicated to its implementation, a decision was made to discontinue its utilisation for this project.

4.2.5 TeachNET

As TeachNET is a complete end-to-end framework, it supersedes the entire algorithm structure employed in the previous algorithm. However, OpenPose was not utilised this time since the input of TeachNet, as referred to in Section 3.3.4 consists of depth images. To obtain the required algorithm input, two cameras were tested: the StereoLabs ZED v1 camera and the Microsoft Kinect v2.

The ZED is a stereo camera developed by StereoLabs that captures high-definition images with depth in real time. The camera utilises two sensors with a baseline of 12 centimetres to

mimic the human stereoscopic vision, enabling it to generate a depth map for each pixel in the image. The camera has a resolution up to 2K (2208 x 1242 pixels) and a field of view of 110 degrees horizontally and 60 degrees vertically. Physically, the camera features a compact and lightweight design, making it suitable for various applications in the field of robotics. The ZED SDK's camera software provides access to the camera features and enables developers to easily interact with the camera functionalities. Moreover, the ZED ROS Wrapper allows the utilisation of the camera in a ROS environment, providing information such as left and right images, a depth map, a coloured depth cloud, pose tracking and others.

The Microsoft Kinect v2 is a 3D camera developed by Microsoft for use with Xbox One gaming consoles and computers. The device integrates a FullHD (1920 x 1080 pixels) RGB camera and a depth camera with a 512 x 424 pixels resolution. The Kinect utilises time-of-flight (ToF) technology to capture depth. This method measures distances based on, as the name implies, time-of-flight, i.e., the round trip time of a light signal emitted by the device. This camera includes a range of features and capabilities that make it suitable for use in various areas, including robotics. It can also be used for full-body motion capture and gesture recognition, allowing immersive gaming experiences. In addition, the Kinect also provides software, namely Kinect for Windows SDK, that enables developers to create applications that support gesture recognition. Moreover, there is a Kinect Bridge ROS package that integrates Kinect and ROS environment.

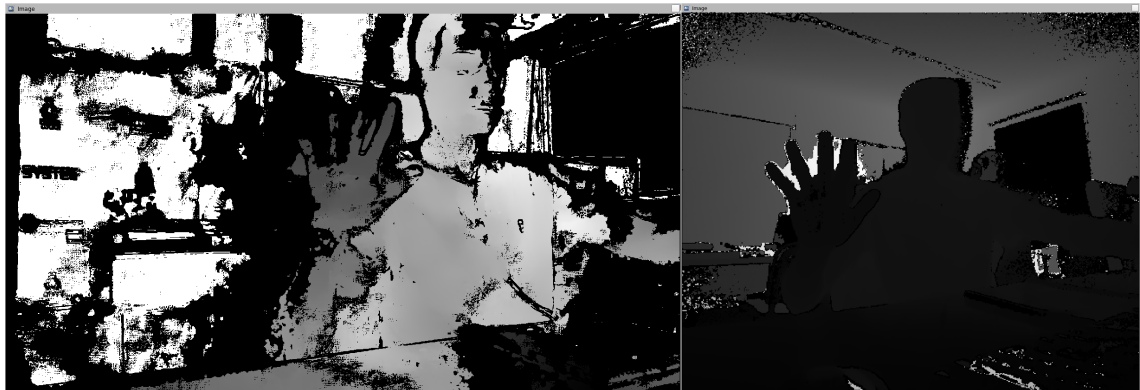


Figure 4.12: ZED (left) and Microsoft Kinect v2 (right) depth images

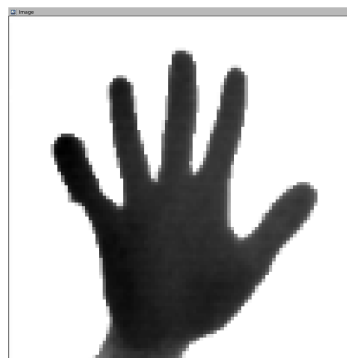


Figure 4.13: TeachNET filtered image (acquired with Microsoft Kinect v2)

Testing the two cameras, although the better resolution on the ZED camera, the Kinect proved superior in capturing depth images. Mainly in matters of stability, the depth images from ZED were very unstable, which made it very difficult to utilise the camera with TeachNET. The quality difference of the images acquired from both images is presented in Figure 4.12. Observing these figures, it is clear to conclude that Kinect performs better. Considering this, the chosen camera was Kinect. In Figure 4.13, it can be seen the filtered human hand from the depth image from Kinect. The utilised filter was part of the initial step of the TeachNET algorithm. Having as input this filtered image, the network outputs joint angles that were subsequently passed to the Shadow Hand in a similar way to the other described methods, thus completing the whole system from the human hand to the Shadow Hand.

The complete architecture is presented in the UML sequence diagram in Figure 4.14.

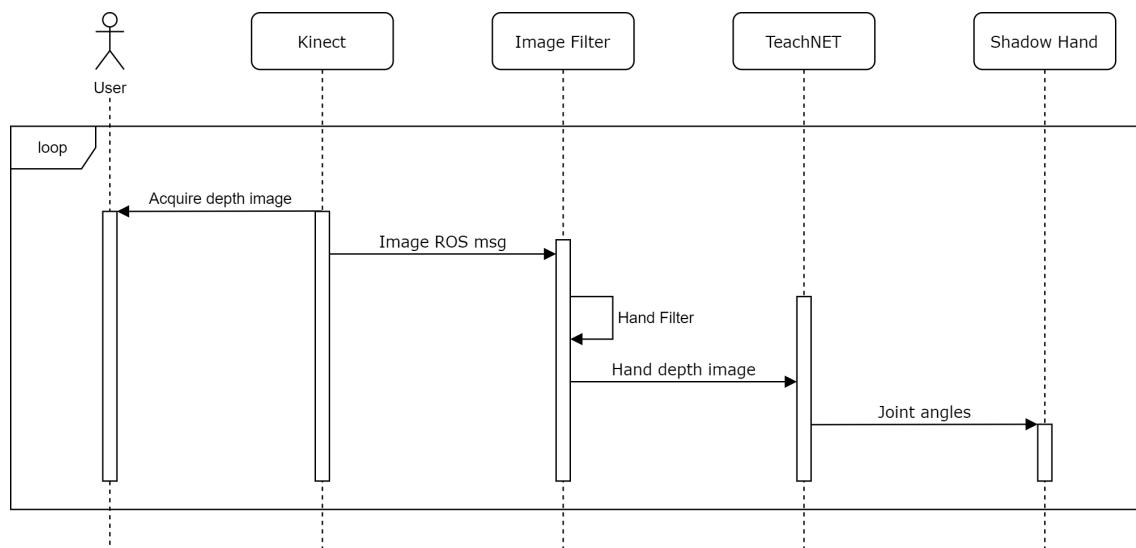


Figure 4.14: Program architecture represented in a UML sequence diagram

Chapter 5

Shadow Hand Control Results

In this chapter, it will be presented and discussed the results achieved through the methods presented and described in Chapter 4, applied to the Shadow Hand.

5.1 MuJoCo Simulation versus Real-World Control

Simulation has become an invaluable tool for designing and testing control strategies due to advantages such as the associated logistical simplicity since the simulation replaces the need for hardware, and, especially in a simulation environment, risks associated with inadequate control, such as premature hardware degradation, are prevented. However, it is crucial to examine how well the simulation replicates the behaviour of the physical system. Therefore, in this chapter, a comparative analysis will be conducted between the control of the Shadow Hand in simulation and in the real-world in order to shed light on the key differences observed in these two scenarios. By uncovering these disparities, it is possible to gain insights into the challenges faced in bridging the gap between simulated and real-world robotic control, ultimately adjusting the control algorithms to achieve a more robust and efficient control intended for real-world applications.

5.1.1 Joint Coupling

In Chapter 4, the described algorithms implementation already accounted for the coupling of Shadow Hand Joint 1 and 2, namely in the DexPilot algorithm with the introduction of the fictitious Joint 0 and in the BioIk algorithm, with the addition of joint function goals. However, these constraints were not considered in the earlier versions of the tested algorithms. The true impact of the coupling was only perceived when transferring the algorithms to the physical hand since, in simulation, Joint 1 and Joint 2 behaved differently.

To highlight the disparities, a series of tests were conducted, taking into account the robotic hand in a simulation environment and in a real environment, before and after the integration of the algorithms part responsible for taking into account the coupling of the distal and proximal interphalangeal joints.

During the performed tests, in order to ensure that the hand pose dissimilarities came only from the actual differences between the simulation and the real robotic hand, a rosbag¹ was recorded, with a fixed position of the human hand, seen in Figure 5.1, acquired by OpenPose. Later, during the tests performed, this rosbag was played in a loop, replacing this way the OpenPose and ensuring that the data acquisition part of the human hand remained constant throughout the experiments.



Figure 5.1: Human Hand position for robotic hand replication reference

Starting with the DexPilot algorithm, Figure 5.2 shows the differences between the replication of the human hand position by the robotic hand in simulation and in reality, utilising the DexPilot algorithm and before the addition of the fictitious Joint 0. Figure 5.3 presents the differences under the same conditions but after Joints 1 and 2 were replaced with Joint 0.



Figure 5.2: Hand pose replication in simulation (left) and in reality (right) by DexPilot before taking into account the joints coupling

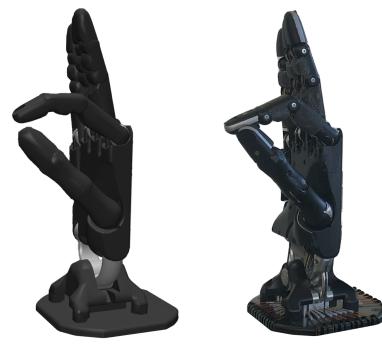


Figure 5.3: Hand pose replication in simulation (left) and in reality (right) by DexPilot after taking into account the joints coupling

Prior to the inclusion of Joint 0 and considering Joints 1 and 2 as independent, it is clear to observe in Figure 5.2 the discrepancy between the distal interphalangeal joint in simulated and in the real hand, as already expected. After the addition of Joint 0, it is possible to notice, in Figure 5.3, that the similarity between the hand position in simulation and in the real-world increased. However, an interesting point to be noted is that despite being given a command of 0 degrees to Joint 1, in simulation, this joint presents some flexion which suggests the existence

¹rosbag: file format utilised in ROS to store and replay recorded data from ROS topics.

of some relationship between the joints, possibly a distribution of Joint 2 angle by Joint 1 and 2, since the forefinger Joint 2 in the real hand seems to be more flexed than in the simulation. The most important thing to underline is that, in fact, the algorithm now considers the actual behaviour of the robotic hand in reality.

Proceeding to the BioIK algorithm, Figure 5.2 shows the differences between the human hand position replication by the robotic hand in simulation and in reality, utilising the DexPilot algorithm as a kinematic solver and before the addition of the constraints to deal with the joints coupling. Figure 5.3 presents the differences under the same conditions after integrating the mentioned constraints.



Figure 5.4: Hand pose replication in simulation (left) and in reality (right) by BioIK before taking into account the joints coupling



Figure 5.5: Hand pose replication in simulation (left) and in reality (right) by BioIK after taking into account the joints coupling

Commencing with the analysis of Figure 5.2, it is possible to observe that it can be seen that both positions are very similar, even verifying the flexion of Joint 1 in the real hand. However, this flexion only occurs due to the force exerted by the forefinger's tip on the thumb's tip. Thus the force imposed by the tendon responsible for driving Joints 1 and 2 is distributed over these two joints. Considering this, the observed hand pose replication was good as there was contact between the fingers; however, for a different scenario, the angle of Joint 1 would not be replicated in reality, except for the complete flexion of Joint 2.

Analysing Figure 5.3, it is possible to verify that, with the inclusion of the additional constraints, the position of the robotic hand slightly deviated from the position of the real hand; however, it is also possible to notice that the algorithm already adapts to the coupling of the joints, resulting in a null flexion for Joint 1. Interestingly, a slight flexion of Joint 1 is observed in the simulation, as has already been observed for the DexPilot algorithm, reinforcing the conclusions previously drawn.

To better understand the actual coupling behaviour existing in simulation and in the real hand, a concise test was conducted. This test consisted of sending different angle values to Joint 1 and 2 of the forefinger and observing the resulting hand position in simulation and the real Shadow Hand.

Between each iteration, the hand was restored to the open hand position in order to ensure the same initial conditions between tests. The results obtained are shown in Table 5.1. Visual results

from the experiment are presented in Figure 5.6, comparing the behaviour of the robotic hand in simulation and in the real-world to the same joint commands.

Table 5.1: Joints coupling test quantitative results in MuJoCo and in the real Shadow Hand

Joint	Sent		MuJoCo			Shadow Hand		
	Angle	FFJ0	Angle	Percentage	FFJ0	Angle	Percentage	FFJ0
FFJ1	45.0°	45.0°	21.6°	48.0%	45.0°	1.5°	3.3%	44.9°
FFJ2	0.0		23.4	52.0%		43.4	96.7%	
FFJ1	0.0°	45.0°	21.6°	48.0%	45.0°	1.5°	3.3%	44.9°
FFJ2	45.0°		23.4°	52.0%		43.4°	96.7%	
FFJ1	45.0°	90.0°	43.1°	47.9%	90.1°	2.7°	3.0%	89.8°
FFJ2	45.0°		46.9°	52.1%		87.1°	97.0%	
FFJ1	90.0°	90.0°	42.3°	47.0%	90.0°	2.7°	3.0%	89.8°
FFJ2	0.0°		47.7°	53.0%		87.1°	97.0%	
FFJ1	0.0°	90.0°	42.3°	47.0%	90.0°	2.7°	3.0%	89.8°
FFJ2	90.0°		47.7°	53.0%		87.1°	97.0%	
FFJ1	90.0°	135.0°	63.5°	47.1%	135.0°	38.8°	28.8%	134.6°
FFJ2	45.0°		71.5°	52.9%		95.8°	71.2%	
FFJ1	45.0°	135.0°	63.6°	47.1%	135.0°	38.8°	28.8%	134.7°
FFJ2	90.0°		71.5°	52.9%		95.9°	71.2%	
FFJ1	90.0°	180.0°	89.9°	50.0%	179.9°	82.2°	45.8%	179.6°
FFJ2	90.0°		90.0°	50.0%		97.4°	54.2%	



Figure 5.6: Joints coupling test qualitative results in MuJoCo and in the real Shadow Hand

The first point to be inferred is that the control of Joints 1 and 2 is, in fact, performed by a fictitious Joint 0 that corresponds to the combined summation of the angles of these two joints since the results are only dependent on the total sum of the angles and not on their distribution between the joints.

The second conclusion is the type of coupling present in the simulation. The results show that the amplitude of Joint 0 is approximately evenly distributed between the two joints.

The findings regarding the real hand reaffirm the previously drawn conclusion, specifically that Joint 1 only exhibits flexion when Joint 2 reaches its limit, and until that point, its flexion is almost unperceived, reaching a maximum of 2.7 degrees. Another interesting fact to extract is that

the actual limit of Joint 2 is slightly higher than expected, reaching 97.4 degrees. Consequently, the maximum angle of Joint 1 is limited to 82.2 degrees.

Summarising, the actual relationship between Joints 0 and 1 has been empirically identified, both in simulation and in reality, and the DexPilot and BioIK solvers were appropriately adapted, leading to the desired outcomes. As a result, improved performance of these algorithms is now anticipated when employed with the physical Shadow Hand.

5.2 Teleoperation

This section presents the experimental results, mainly in a visual format, of the application of the various kinematic retargeting techniques, namely direct joints mapping, DexPilot, BioIK and TeachNET, respectively.

For the test of the first algorithm, considering that it consists of a very preliminary version aimed at verifying the complete integration from OpenPose to Shadow Hand, a simple experiment was conducted. This experiment consisted of individually and successively closing and opening each one of the fingers of the hand.



Figure 5.7: Clips utilised for the wiring fitting task

For the remaining three algorithms, the test was performed with the clipping goal already in consideration. Figure 5.7 presents the clips to be utilised. As it is possible to observe, these clips are small, and their manipulation involves a precise grasp. Thus, the tests performed involved replicating a pinching grasp position, possibly applicable to the actual context of the wiring fitting.

Each experiment started with an open and stable hand posture, and then a pinching grasp was performed and maintained briefly. In this way, it was possible to verify the quality of replication of the grasp position and the stability provided by each algorithm.

The experiments were recorded with the human hand positioned adjacent to the Shadow Hand to facilitate a comprehensive comparison of hand position replication. The results to be presented comprise frames extracted from the recorded videos at significant timestamps.

5.2.1 Direct Joints Mapping

The direct map of joint angles from the human hand to the Shadow Hand resulted in the gestures replication shown in Figure 5.8.



Figure 5.8: Direct joints mapping algorithm test results

Figure 5.8 illustrates the progression from a fully opened hand to different finger configurations. By comparing the displayed images, it is perceptible that the algorithm worked effectively by translating the position of each finger in the human hand to the Shadow Hand.

Furthermore, the results of the conducted experiment validate the successful integration of the end-to-end algorithm for reproducing human hand movements on the Shadow Hand.

5.2.2 DexPilot

As previously mentioned, in order to test the DexPilot algorithm, a pinching movement was performed with the index finger and thumb. Figure 5.9 presents the different frames from the experiment, demonstrating the human hand movements and the respective replication by the robotic hand. Each frame has the respective timestamp annotated.

Upon analysing the results, in the first place, it is possible to perceive the correct functioning of the algorithm by the replication of the movements. Further analysis shows the stability of the hand positions generated by DexPilot, namely in the initial and final parts of the experiment. Despite this, it is possible to verify that the pinching position is not replicated precisely. This happens due to the intrinsic characteristics of the algorithm, explained in 4.2.2, namely that, with the fingers being at a distance inferior to the ε_1 threshold, they are forced to a position where the distance between them is minimal, defined by η_1 . This way, the algorithm forces the closing distance between the fingers without prioritising their relative position.

Even though the position is not exactly replicated, the degree of repeatability of DexPilot is high among different tests, as seen in Figure 5.10, which is a great advantage of this algorithm.

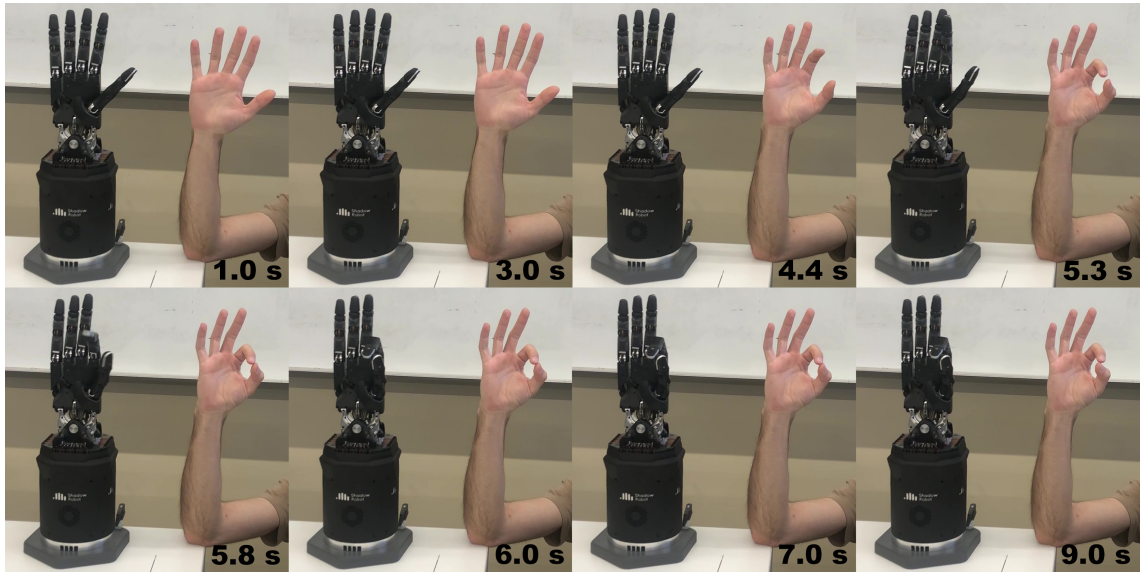


Figure 5.9: DexPilot: experimental results



Figure 5.10: DexPilot: pinching pose repeatability

To better comprehend and analyse the algorithm's operation, the graphic presented in Figure 5.11 shows the distance between the tip of the forefinger and the thumb in both the human hand and the robotic hand. The measures relative to the human hand were acquired from OpenPose, after the application of a median filter; on the other hand, the Shadow fingertip distances were raw acquired from the Hand ROS interface, utilising the ROS Transform System (TF). The ROS TF is a package that keeps track of all the robot referential frames, for example, in this case, the fingertips, the wrist and other articulations that describe the robot model. Furthermore, TF facilitates working and performing operations between the different existing frames. For this specific case, in order to calculate the distance between the fingertips, TF was utilised to obtain the translation vector between the thumb tip frame and the forefinger tip frame. Then the distance was directly given by the magnitude of this vector. To refer that, these two measures, from the human and the robotic hand, were acquired at the same time in the thread responsible for the DexPilot solver before calling it, ensuring this way that the values acquired and shown in the graphic in Figure 5.11 were the same input to the algorithm.

Analysing the graphic, the refereed stability of the robotic hand is confirmed by the stability of the red line. Another important aspect to extract from the graphic is the delay between the human

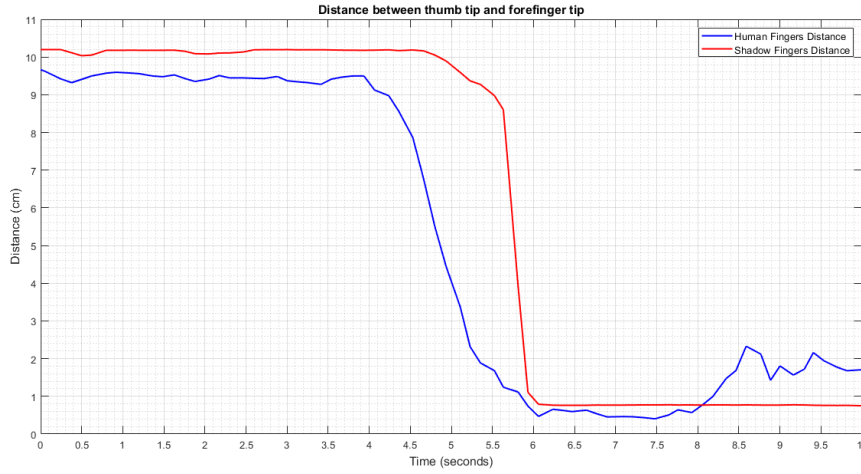


Figure 5.11: DexPilot: distance between thumb tip and forefinger tip

hand pose acquisition and the respective actuation on the robotic hand. Considering the properties of this algorithm, it is possible to define this delay as the time interval between the moment when the distance between the fingers of the human hand reaches the threshold ε_1 and the moment when the fingers of the robotic hand start the pinching movement. Given that ε_1 was defined as 3 centimetres in 4.12 and observing the graphic in Figure 5.11, it is possible to infer that the first described moment occurs at 5.2 seconds. The second moment is defined by the abrupt decrease of the red line, which occurs at 5.6 seconds. It is thus possible to conclude that there is a delay of about 400 milliseconds. This delay includes the running time of the algorithm itself in addition to the time of sending and executing commands by Shadow Hand. The algorithm execution time was verified and corresponded to an average of 56 milliseconds. This leads to the conclusion that most algorithm delay time is related to the sending and execution of commands. In addition to these times, the total delay of the algorithm also includes the delay intrinsic to OpenPose and the delay introduced by the application of the median filter. When considering these factors, the overall delay of the system amounts to approximately 1 second.

It should be noted that the execution times of the algorithms were measured on a computer with the following specifications:

- Central Processing Unit (CPU): AMD Ryzen™ 9 7950X3D
- Random Access Memory (RAM): 32.0 GB
- Graphics Processing Unit (GPU): NVIDIA GeForce RTX 4090 24GB

5.2.3 BioIK

The BioIK algorithm was also tested using the same experiment, and the results are presented in Figure 5.12 as a sequence of frames.

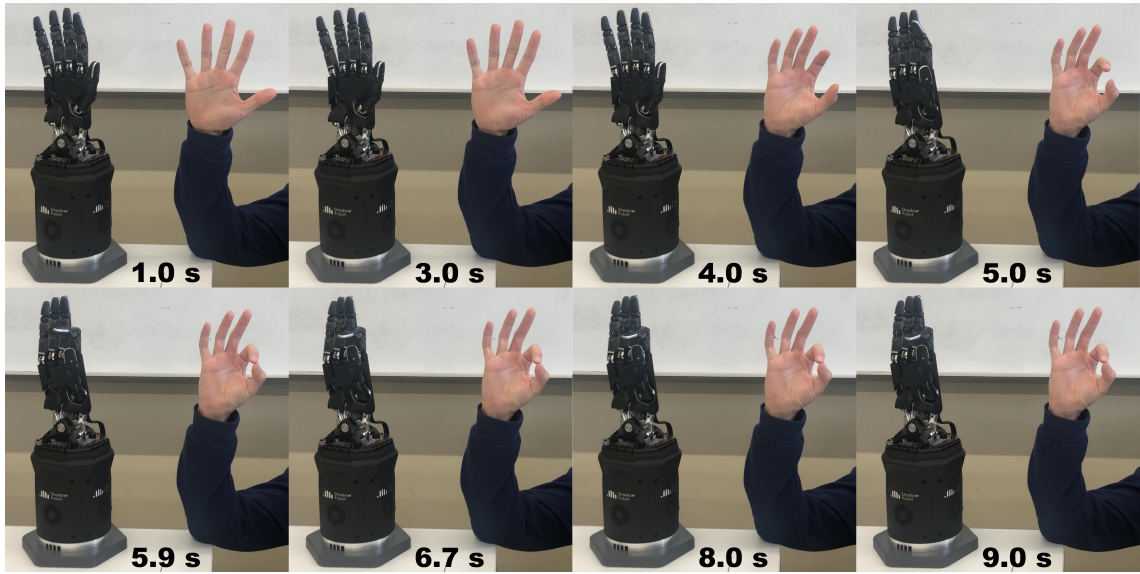


Figure 5.12: BioIK: experimental results

Regarding the results obtained, it is possible to verify that the hand position is replicated more effectively; however, there is a relatively large gap between the fingers, which could represent a problem for the application in question since the clip may not be adequately gripped. In addition, the degree of repeatability achieved is inferior compared to DexPilot, as shown in Figure 5.13.



Figure 5.13: BioIK: pinching pose repeatability

In addition to the visual results, data about the fingertips' positions were acquired in the same way as already explained for the DexPilot algorithm and utilised to make two graphics. The first one, presented in Figure 5.14, shows the distance between the thumb and forefinger tips, and the other, presented in Figure 5.15, shows the distance between the human fingertips and the respective robotic fingertips since this was the main goal specified for this algorithm. In this last one, a moving average filter was applied to smooth the data.

By the analysis of the graphic presented in Figure 5.14, it is possible to perceive, by the red line, the instability of the poses generated by BioIK, when compared to DexPilot, presented in Figure 5.11.

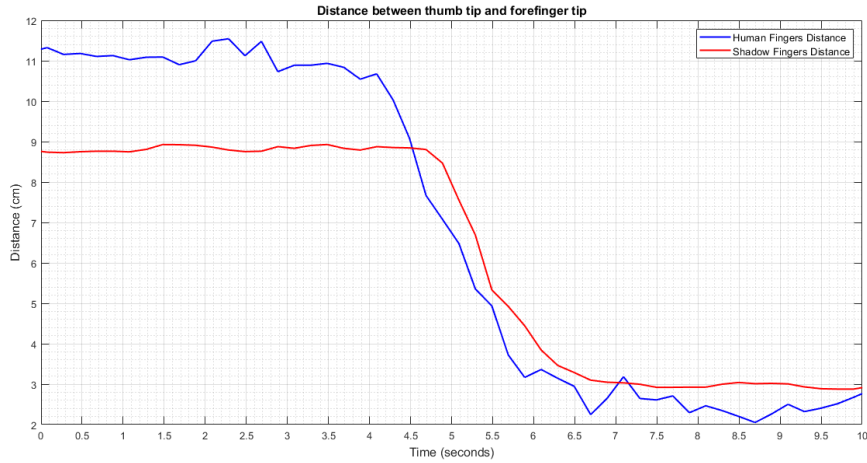


Figure 5.14: BioIK: distance between thumb tip and forefinger tip

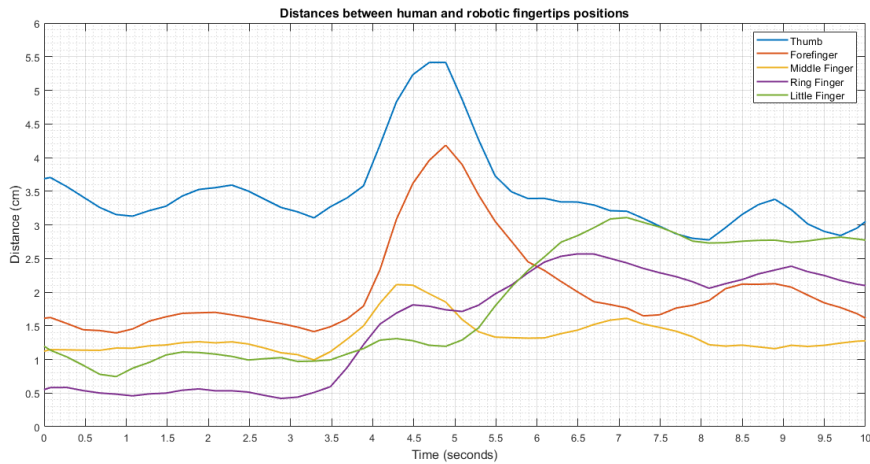


Figure 5.15: BioIK: distances between human and robotic fingertips positions

Analysing the graphic presented in Figure 5.15, it is possible to observe the moment where the pinching movement starts, which corresponds to the moment where the thumb and forefinger tips distances from human to robotic hand starts to increase, which occurs at about 4 seconds. At about 5 seconds, it is possible to verify the algorithm actuation by the reduction of these positioning differences. Nevertheless, the final position differences are considerably large, varying between 1.3 and 3.1 centimetres.

Regarding the time delay, given that the structure is the same as the DexPilot algorithm, the only difference lies in the running time of the algorithm itself, which was verified to be 216 milliseconds, almost four times higher than the solving time of DexPilot.

5.2.4 TeachNET

Lastly, the teleoperation results achieved with TeachNET will be shown.

Primarily, and to better understand the end-to-end TeachNET algorithm, Figure 5.16 presents a sequence of 4 images describing the TeachNET process. First, the desired hand position is performed, and the Kinect acquires a depth image of the human hand. Then, the hand is filtered from this image and given as input to the TeachNET network. Lastly, the network outputs the joint angles to control the robotic hand, resulting in the position in the rightmost picture shown in Figure 5.16.

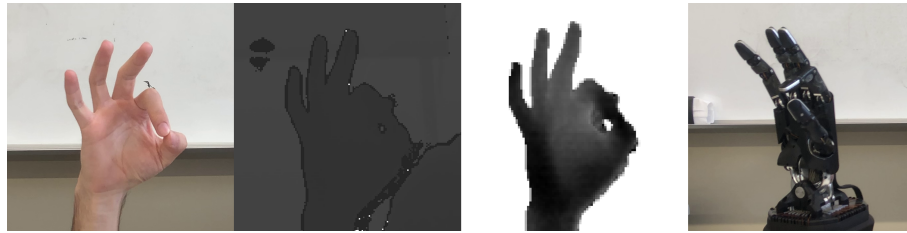


Figure 5.16: TeachNET process: human hand pose, Kinect depth image, TeachNET segmented image, robotic hand pose, from left to right, respectively

The results of the teleoperation are visually presented in Figure 5.17

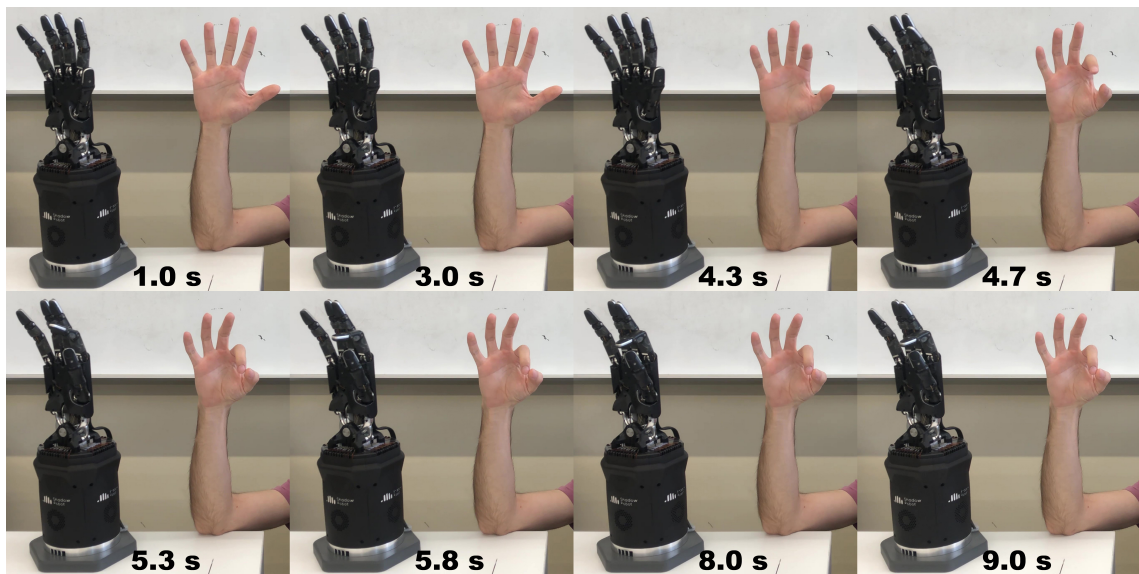


Figure 5.17: TeachNET: experimental results

From the experiment, it was concluded that TeachNET results in a stable operation of Shadow Hand; however, the replicated position does not have a high degree of repeatability, as proven by the results shown in Figure 5.18, this is, for the same grasp position of the human hand, slightly different grasp positions in the robotic hand were obtained. Considering this, sometimes the resulting gap between the pinching fingers was too large, considering the clips' dimensions, which could be a problem for the wiring fitting task.

Regarding the delay, this algorithm behaved best, with a time of about 300 milliseconds between the movement of the human hand and its replication by the Shadow Hand.



Figure 5.18: TeachNET: pinching pose repeatability

Concluding, TeachNET proved to be a good overall algorithm for teleoperation; however, it may not be the most adequate for precision grasps.

5.3 Algorithms Comparison

Considering the evaluation of the kinematic retargeting algorithms, it is already possible to extract some findings. Regarding control stability, during the tests, it can be noted that the most stable algorithm is TeachNET, followed by DexPilot, thus leaving BioIK as the least stable among the three algorithms. Another essential aspect remains in replicating the pinching grasp, namely the gap distance between the fingers. In this aspect, DexPilot undoubtedly stands out by ensuring that the fingers are touching. On the other hand, when utilising the remaining algorithms, this distance is not always the most appropriate, and sometimes there are relatively large gaps between the fingers, resulting in an inability to grasp the clip.

To obtain a more effective and quantitative comparison between the three kinematic retargeting algorithms, a practical test based on the real scenario of the application task of the Shadow Hand was conducted. The main goal of this practical experiment was to show the controllability skills provided by each of the algorithms studied in a real-case scenario and its utilisation by regular users unfamiliar with this type of control. In addition, this test allowed to understand the robotic hand's actual dexterity level.

Description of the conducted practical algorithms test:

- Main task: hold the cable [Figure 5.7] by a clip;
- Preparation: each of the participants was briefly informed in advance of the task to be performed and how each of the algorithms works;
- Train: each participant had a total training time of 6 minutes for each algorithm. For the first 3 minutes, training was conducted with the robotic hand only so that each participant could adapt to the robotic hand movements resulting from each algorithm. During the remaining 3 minutes, the training was performed with the cable, with the goal of training a hand position capable of holding the cable by a clip;

- Task execution: first, the cable was positioned close to the robotic hand, within its limits, so that it could be held; afterwards, the goal was to hold the clip with the robotic hand firmly; finally, the external force holding the cable was removed, in order to see if the robotic hand was able to hold the cable suspended;
- Attempts: the task was repeated, by each participant, a total of 15 times, 5 for each clip shown in Figure 5.7. The order of the clips was random;
- Score:
 - 0 points: the cable was not gripped at all, or the cable slipped as soon as the external force holding it was released;
 - 1 point: the cable was held for a period of less than 2 seconds;
 - 2 points: the cable was held for a time greater than 2 seconds and inferior to 5 seconds;
 - 4 points: the cable was held for a time greater than 5 seconds;
- Number of participants: 10 participants;
- Note: the order of the algorithms tested for each of the participants was randomly determined.

From the tests performed, the number of participants (corresponding to the respective participation order), the name of the algorithm used, the trial number, and which clip was used were noted for each trial. These individual results are present in Appendix A. The following results include the performance of each of the participants relative to each algorithm [Figure 5.19], to each clip [Figure 5.20], to each attempt [Figure 5.21] and finally the overall performance is presented in Figure 5.22. Global results were also extracted on the performance achieved with each of the algorithms [Figure 5.23], performance per clip [Figure 5.24] and performance per attempt [Figure 5.25]. Finally, these three graphs were repeated, however, taking into account the percentage of success, i.e., the percentage of attempts whose score was maximum, in Figure 5.26, Figure 5.27 and Figure 5.28, respectively. Furthermore, qualitative results were extracted, taking into account successful grasps. These results are present in Figure 5.30.

Analysing the graphic in Figure 5.19, which presents the relation between each tested algorithm and the average score achieved by each participant, it is possible to verify that for all individuals, the BioIK algorithm translates into the lowest performance obtained. Also noteworthy is the DexPilot algorithm, responsible for the maximum average score of 80% of participants. The TeachNET algorithm lies between the two mentioned, yet, always very close to DexPilot.

Evaluating now, the information presented in the graphic of Figure 5.20, that relates the clip with the achieved score by each participant, it does not seem to be a constant correlation between these two factors; perhaps there is overall a better gripping of clip 2, to be confirmed by the analysis of the global results.

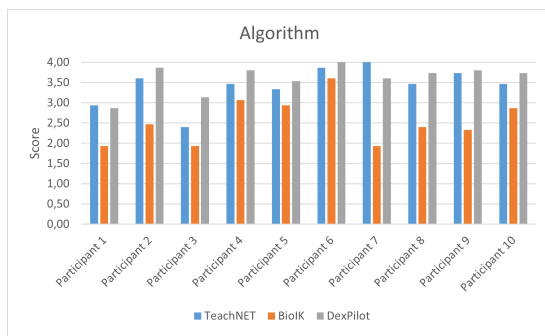


Figure 5.19: Performance of each participant by algorithm

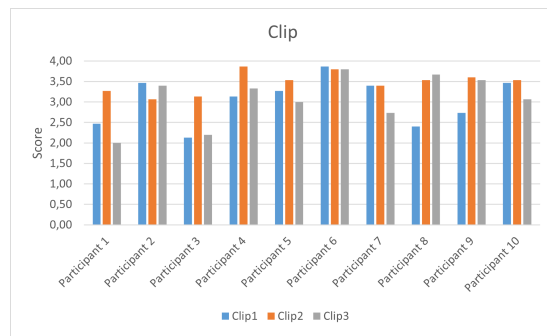


Figure 5.20: Performance of each participant by clip

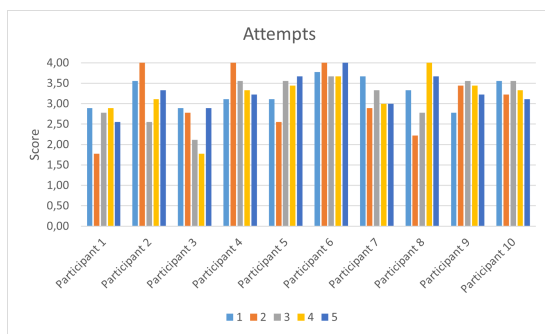


Figure 5.21: Performance of each participant by attempt

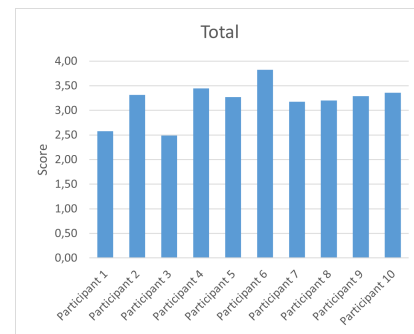


Figure 5.22: Global performance of each participant

The graphic presented in Figure 5.21 shows the relationship between each participant attempt and the respective average score obtained. This graphic was made with the goal of verifying if the training time was adequate. For example, if the average score increased with the attempt number, it could mean that the training time was insufficient; however, examining the data in the graphic, it is not possible to extract a pattern, which possibly means that the training time was adequate.

The global score of each participant, present in the graphic in Figure 5.22, remained between 2.49 and 3.82, with 70% of average scores between 3.00 and 3.50.

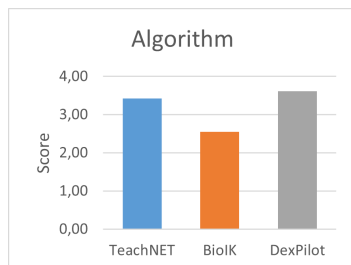


Figure 5.23: Global performance by algorithm

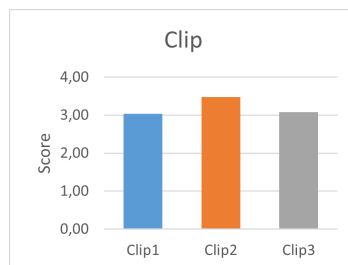


Figure 5.24: Global performance by clip

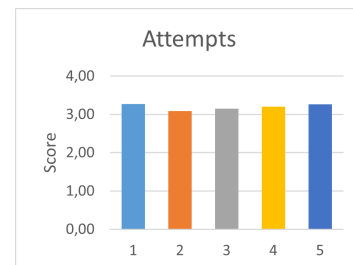


Figure 5.25: Global performance by attempt

Moving to the global results analysis, the previously drawn conclusions are confirmed. The algorithm with the best performance is DexPilot, followed by TeachNET, with BioIK ranking last,

as can be observed in the graphic shown in Figure 5.23. Regarding the clips, it is possible to infer from the graphic in Figure 5.24 that clip 2 has a slightly higher score, which is curious, since this is the smallest and supposedly most challenging clip to grab. Clips 1 and 2 remain on par. Lastly, the graphic in Figure 5.25 demonstrates an equilibrium between the attempt number and the respective achieved score, thus making it clear that there is no correlation between these two factors.

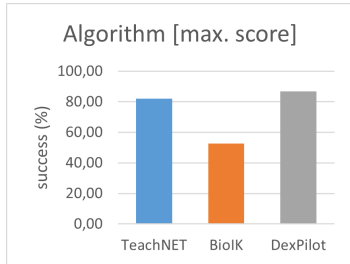


Figure 5.26: Global maximum score attempts by algorithm

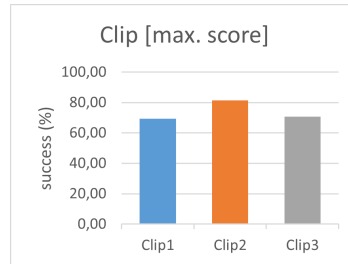


Figure 5.27: Global maximum score attempts by clip

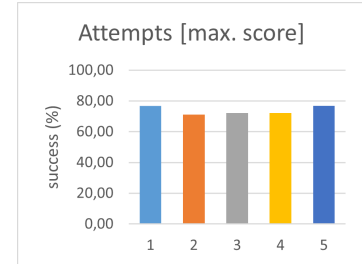


Figure 5.28: Global maximum score attempts by attempt

From the graphics shown in Figures 5.26, 5.27 and 5.28, which present the percentage of attempts in which the maximum score was achieved, by algorithm, clip and attempt number, respectively, the previously drawn conclusions are confirmed. Thus, these results are intended to reinforce the earlier findings.

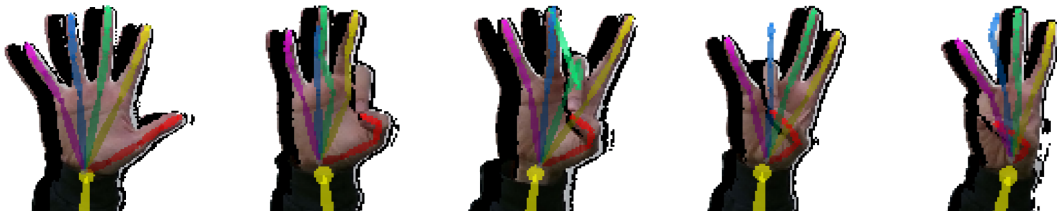


Figure 5.29: OpenPose misdetections (left image for reference)

In addition to these results, during the experimental Shadow Hand teleoperation tests, it was also possible to verify, also due to comments made by the participants, that one of the significant limiting factors of the BioIK algorithm resided in the acquisition of the human hand position by OpenPose, being frequently unstable and inaccurate. Frequently, OpenPose incorrectly detects some of the fingers, as can be seen in Figure 5.29 by the change in the characteristic detection colour of each finger. Furthermore, even with an apparently correct detection, it is enough for a keypoint to be detected slightly off the real position for the corresponding depth to be quite different from the real depth of the keypoint. For example, if a fingertip keypoint is detected slightly off the finger, the corresponding depth will be the background depth, leading to large errors, as previously seen in Figure 4.2.

To conclude, the qualitative results of the tests are presented in Figure 5.30. This figure shows successful clip grasps from each algorithm and clip. The results are organised in the following



Figure 5.30: Successful grasps from tests. The rows, from top to bottom, correspond to the DexPilot, BioIK and TeachNET algorithms, respectively. The left to right columns correspond to clips 1, 2 and 3, respectively.

way: DexPilot in the first row, BioIK in the second row and TeachNET in the third row; clip 1 in the first column, clip 2 in the second column and clip 3 in the third column.

Due to the intrinsic characteristics of the algorithm, DexPilot achieves a more natural human hand grip position, as it is possible to observe in the top row. For the remaining algorithms, some “tricks” are required to ensure that the robotic hand’s position is appropriate to grasp the clip and with sufficient force to hold the cable by the clip. Specifically, it is possible to notice in the majority of the attempts, the forefinger is placed over the thumb to enforce contact between them in the robotic hand. Consequently, it becomes more challenging for the participants to find and replicate the ideal position to grasp the clip, resulting in a lower effectiveness of these algorithms, as already verified through the analysis of the quantitative results. The participants also mentioned this difficulty during the experiment, especially during the BioIK algorithm test.

Upon careful analysis of the test results, it can be deduced that the DexPilot algorithm is the most suitable for the given task. However, the Shadow Hand, in the gripping position generated by this algorithm, does not have the necessary tendons tension to properly execute the clipping task. When force is exerted on the clip, the joints concede, causing the clip to slip through the fingers. It is thus possible to conclude that the accomplishment of the clipping task is not achieved using this algorithm or other of the algorithms mentioned, and therefore another strategy must be pursued.

Chapter 6

Implementation of Wiring Fitting and Paint Quality Inspection Tasks

This chapter encompasses the practical implementation of the two principal tasks proposed within the context of this dissertation.

6.1 Wiring Fitting

This section presents the undertaken efforts aligned with the objective of accomplishing the wiring fitting task.

6.1.1 Developed Algorithm

Drawing upon the comprehensive knowledge gained from Chapter 5 about OpenPose, the different algorithms tested, and the dexterity and strength of the robotic hand by the several tests conducted, a novel algorithm for the wiring fitting task was designed and conceived to take into account and potentially overcome all the current existing limitations.

This newly developed algorithm relies on the state machine shown in Figure 6.1.

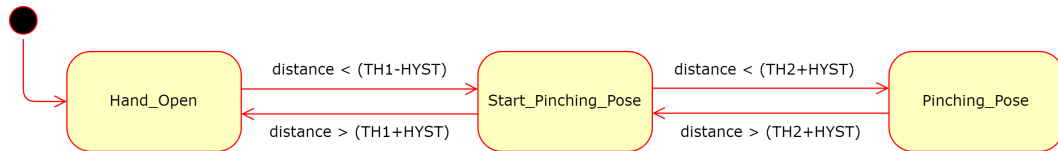


Figure 6.1: UML State Machine Diagram of the developed algorithm

Each of the constituent states of the state machine corresponds to a particular position of the robotic hand. There are a total of 3 states. In the initial state, the hand begins with an open position. The next position corresponds to an approximation position of the pinching grasp so that it is possible to identify how the clip will be held to correctly place it relative to the robotic hand. Lastly, the final state corresponds to the pinching grasp, where the clip is effectively held and ready for clipping.

The *a priori* determination of distinct hand positions aims to address the challenges encountered during testing when attempting to reproduce and maintain a specific position of the robotic hand. This approach circumvents the difficulty of replicating and preserving a particular hand configuration, ultimately guaranteeing the consistency and stability of the hand's position.

The state machine's transitions were defined considering the distance between the forefinger tip and the thumb tip. They depended on threshold values with hysteresis, as presented in Figure 6.1. This way, a transition occurs when a specific condition, determined by a threshold, is surpassed. However, hysteresis was employed to avoid erratic and undesirable transactions caused by slight fluctuations of OpenPose predicted keypoints. Hysteresis introduces a second threshold that must be crossed in the opposite direction to revert the transition and return to the previous state. By incorporating hysteresis, the algorithm ensured stability and robustness by requiring a distinct change in the input values to trigger state transitions, preventing the OpenPose related issues encountered during testing.

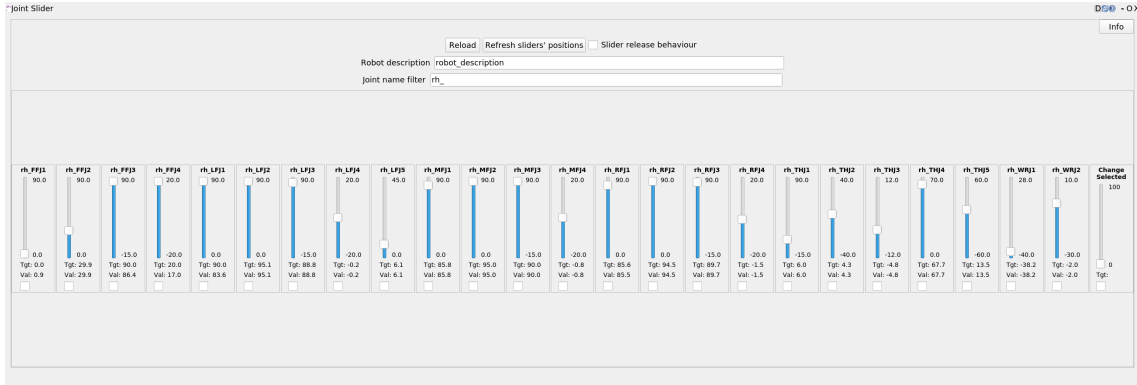


Figure 6.2: Joint Slider plugin

To establish the positions of the hand in the “Start_Pinching_Pose” and “Pinching_Pose” states, it was first utilised TeachNET algorithm to get the initial poses. The generated postures were then adjusted with the “Zero Force Mode” of Shadow Hand, which allows the manual and free movement of the hand joints. In addition, it was also taken advantage of the “Joint Slider” plugin. This plugin provides a graphical interface that allows individual and fine adjustments to each robotic hand joint, as seen in Figure 6.2. These methods were together utilised to achieve the final hand positions, and subsequently, the respective angles of each joint were acquired and saved in order to later replicate these positions in each respective state.

Regarding the value of thresholds and hysteresis, these were defined experimentally as:

$$\begin{aligned} TH1 &= 5.50 \text{ cm} \\ TH2 &= 3.50 \text{ cm} \\ HYST &= 0.50 \text{ cm} \end{aligned} \tag{6.1}$$

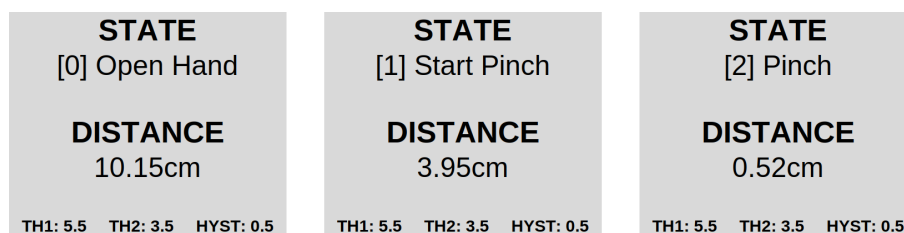


Figure 6.3: Graphical Interface of the algorithm

In order to better comprehend the algorithm's behaviour in real-time, a simple graphical interface was developed, which can be seen in Figure 6.3. This interface includes the name of the current state, the distance between the index finger and thumb tips, and the defined threshold and hysteresis values.

Finally, it only remained to define the hand positions for each state. After several attempts, the positions that showed superior results in terms of holding tight the clip are the ones shown in Figure 6.4, for the states "Hand_Open", "Start_Pinching_Position" and "Pinching_Position", from left to right, respectively.



Figure 6.4: Hand Positions for states "Hand_Open", "Start_Pinching_Position" and "Pinching_Position", respectively

However, despite being the best hand positions found, it was not possible to perform the clipping using the precision grasp seen on the right side of Figure 6.4. Images of the clipping attempt are present in Figure 6.5.

As it is visible, the existing space on the edge of the clip, which allows the application of force for clipping, is too small considering the size of the fingertips of the robotic hand, having only 3 millimetres. This, coupled with a slight slippage of the clip due to the bending of the joints resulting from the force exerted, results in the impossibility of clipping task by the Shadow Hand. Analysing Figure 6.5, it is possible to observe that the fingertips are already touching the car door while the clip is only partially inserted, making it impossible to fully insert the clip.

After a meticulous analysis of the anatomical structure of the robotic hand, it was understood that an out-of-the-box solution that provided the application of the required clipping pressure to the clip was to hold it between the proximal phalanges of the middle and ring finger in a clenched



Figure 6.5: Unsuccessful clipping attempt with the developed algorithm (left, top and right view-points, respectively)

fist position, as it is illustrated in Figure 6.6. In this way, the necessary force for a successful clipping was physically ensured for the fingertips hitting the palm and not entirely dependent on the artificial tendons tension, as it can be observed by analysing Figure 6.7. Additionally, this hand posture securely holds the clip in place since, for the robotic hand, in this position, it is apparently physically impossible to move the phalanges apart, preventing this way any potential slippage of the clip between the fingers.



Figure 6.6: Clenched fist hand positions for states “Start_Pinching_Position” and “Pinching_Position”, respectively



Figure 6.7: Clenched fist hand position holding the clip

An instance of a proficient clipping procedure utilising the aforementioned technique is depicted in Figure 6.8. The sequence of frames displayed illustrates the successive stages of the clipping process.

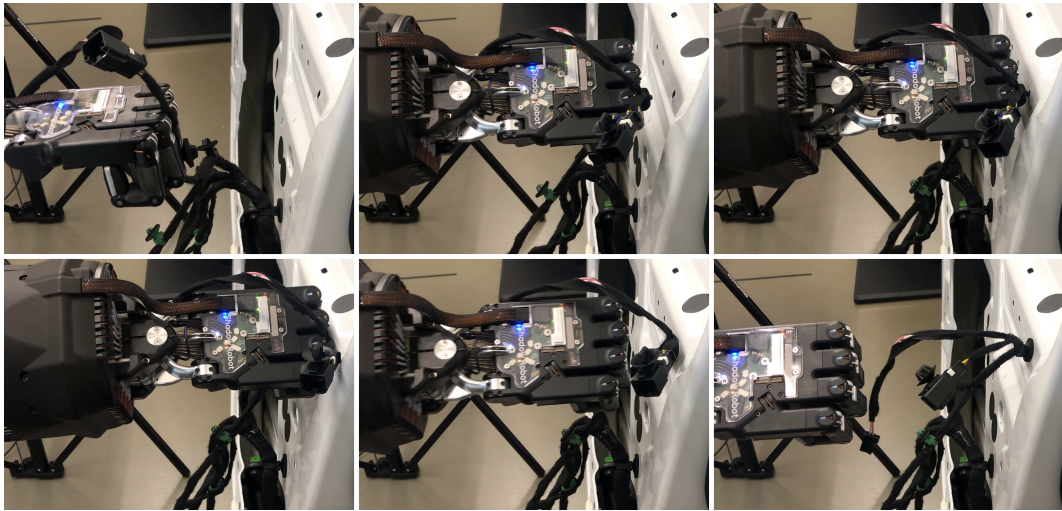


Figure 6.8: Clenched fist hand position performing the clipping task

As observed, the implementation of the clenched fist technique proved to be effective in achieving the wiring fitting task. However, considering that in a manufacturing environment, the clipping task is not performed utilising the proximal phalanges in a clenched fist position, this technique infringes one of the primary purposes of the project where this dissertation is integrated, which is the replication of the human hand movements. Therefore, once again, exploring an alternative approach for accomplishing the wiring fitting task becomes necessary.

6.1.2 Clip Adapter

Considering that the primary challenge for the wiring fitting task was associated with the limited dimensions of the clip, particularly the narrow surface area where the robotic hand needed to exert its clipping force, a viable solution was devised. This solution involved the development of an adapter intended to be attached to the original clip, with the explicit purpose of providing a larger contact area to make the clipping procedure conceivable. Taking into account his goal, the clip adapter was designed and then 3D printed.

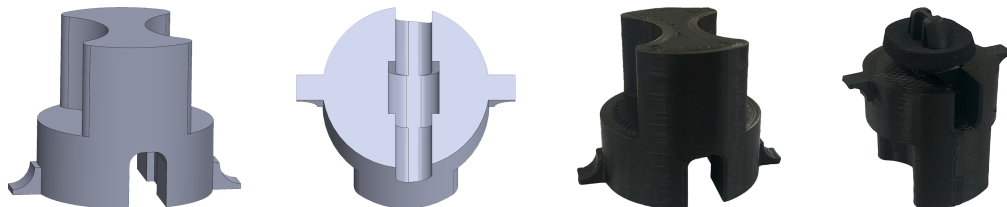


Figure 6.9: Clip Adapter: 3D model (left side) and 3D printed result (right side)

A visual representation of the ultimate iteration of the clip adapter is illustrated in Figure 6.9. It comprises two fundamental components: the upper section designed to serve as a gripping mechanism for the robotic hand and the lower area intended to accommodate the original clip securely. Additionally, the lower section of the adapter also features a transverse cross-section designed to

provide a suitable space for accommodating the cable to which the clip is attached. Lastly, the presence of auxiliary side supports ensures the clip adapter's correct orientation alignment when placed on the initial support structure. Through the utilisation of this adapter, the robotic hand successfully achieved a secure grasp of the clip, as demonstrated in Figure 6.10, while exerting the necessary force for the clipping action.



Figure 6.10: Shadow Hand grasping the clip with the adapter from different angles

An experimental test was conducted to evaluate the clip adapter's effectiveness. The test involved coupling the Shadow Hand to a UR5 robotic arm manually operated through its user-friendly console interface. This interface intuitively controls the robot's movements by providing directional arrows to command its intended direction. The procedure commenced by prepositioning the clip within the fingers of the robotic hand, which assumed a pinch position. Subsequently, the robotic arm was manually guided to the clipping position. Upon insertion of the clip, the Shadow Hand releases it, and the arm is retracted. The entire experiment was recorded, and the results are presented in Figure 6.11, consisting of frames captured at relevant timestamps. The findings demonstrate that the utilisation of the clip adapter effectively allowed the accomplishment of the clipping task.

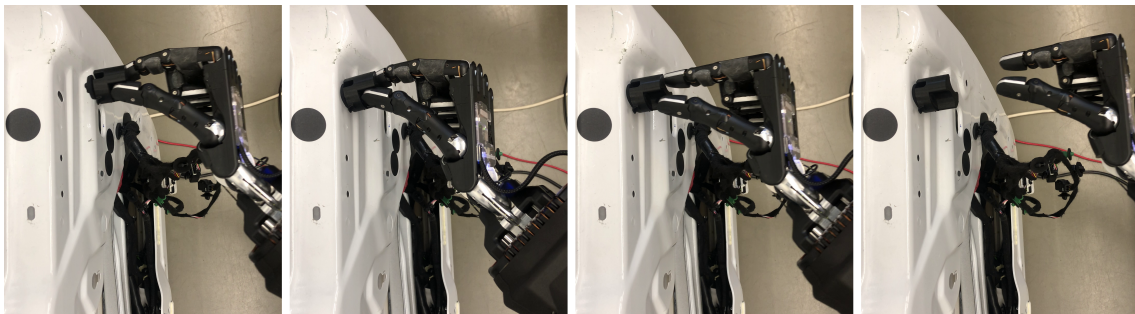


Figure 6.11: Clipping task performance utilising the clip adapter

6.1.3 Experimental Implementation

At this stage, after successfully executing the clipping action, additional steps need to be completed to fully accomplish the task autonomously. These steps involve autonomously picking up the clip, accurately positioning and clipping it in its designated location, as illustrated in Figure 6.12, and then repeating the process for the remaining clips.

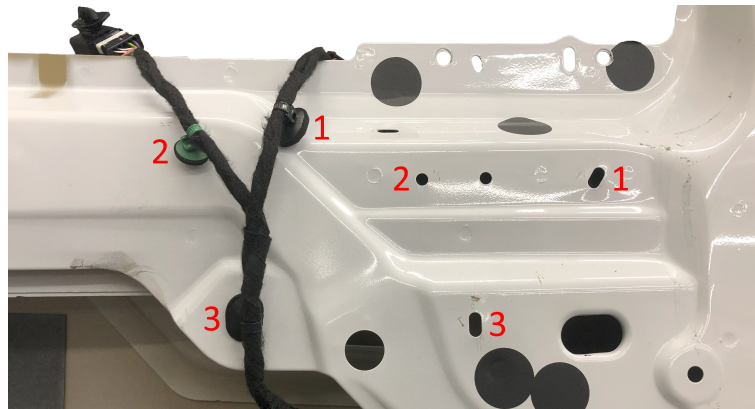


Figure 6.12: Numerically identified clips and corresponding clipping locations

Given that the central focus of the dissertation was on the control of the robotic hand rather than the localisation aspect, the clips, detached from the cable, were initially positioned in a support structure in a predetermined and known position. Similarly, the insertion position for the clips was also assumed to be known.

As demonstrated in Chapter 5, the teleoperation method was found to lack the required level of accuracy for successfully completing the wiring fitting task. Consequently, the application of Learning by Demonstration techniques, as explored in Chapter 3, which involves learning tasks through demonstration, were considered unsuitable. As a result, an alternative approach was pursued.

Considering that the robotic hand control was already established, with the defined poses to firmly grasp the clip, it was decided to control the UR5 robotic arm and the Shadow Hand independently. The sequence of actions for each component, with regard to each clip, can be described as follows:

1. Position the robotic hand in the “Start_Pinching_Position” configuration and position the robotic arm at a distance of 5 centimetres from the clip to the fingertips, in a direction normal to the clip base;
2. Move the robotic arm to attain an optimal position to grasp the clip, and then command the robotic hand to the “Pinching_Position” configuration;
3. Return the robotic arm to the position described in step 1;
4. Position the robotic arm at a distance of 5 centimetres from the clip’s insertion place in a direction normal to the door structure;
5. Move the robotic arm in order to accomplish the clipping task, and subsequently position the robotic hand in the “Start_Pinching_Position” configuration to release the clip;
6. Lastly, return the robotic arm to the position specified in step 4.

Note that the positioning carried out in steps 1, 3, 4 and 5 is intended to ensure an approach and respective return movement in a direction normal to the target.

By following this sequence, the robotic arm and hand can work in tandem to manipulate each clip to accomplish the proposed task successfully. To enhance comprehension of each of these sequential steps, they are visually depicted in Figure 6.13.



Figure 6.13: Wiring fitting task: steps 1 to 6, from left to right, respectively

The control of the robot arm involved providing information regarding the desired position of the tool flange, which corresponds to the location on the UR5 robot where the Shadow Hand was coupled. However, since the positioning objective focused on the fingers of the robotic hand, specifically the forefinger and thumb, for grasping and manipulating the clip, a transformation of the reference frame was required.

Considering that the hand positions during the task execution were well-defined and fixed, two reference frame changes were considered. These conversions involved the “Start_Pinching_Position” position of the robotic hand for grasping the clip and the “Pinching_Position” for the actual clipping task. The transformation process was similar for both cases. For instance, in the case of the clip already held by the robotic hand, the robotic arm was manually adjusted to position the clip in a predetermined pose (position and orientation). Subsequently, the robotic arm’s tool flange position was obtained utilising the ROS TF. With these two known frame references, the conversion between them was computed. This computation allowed the determination of the robotic arm tool flange’s pose based on the clip’s desired pose.

Once the implementation of the algorithm was finished, the next step consisted of its experimental testing. By employing the aforementioned methodology, the designated task was effectively accomplished. However, during various tests, it was noticed that the algorithm was very dependent on the given positions. Considering the dimensions of the clip, a position error of only one millimetre could already result in an unsuccessful clipping. It can therefore be concluded that, ideally, a localisation algorithm and corresponding correction of the real-time position and orientation of the clip tip in relation to its clipping location would be necessary to ensure a high task success rate.

A demonstration video showcasing a successful execution of the task, can be accessed by visiting this [link](https://youtu.be/OPnqiDPKQko)¹.

¹<https://youtu.be/OPnqiDPKQko>

6.2 Automotive Painting Quality Control

Transitioning to the task of automotive painting quality control, the primary objective entails replicating the hand movements executed by a human operator during the tactile assessment of the paint surface, with the aim of identifying any defects or irregularities.

To capture data from the human hand, the MediaPipe framework, in conjunction with a stereo-vision ZED camera, was employed. It is important to note that the ZED camera was utilised solely for capturing raw images from both of its individual cameras, functioning as two independent cameras. However, the fixed and known distance between the two cameras provided an advantageous condition for the application of stereo-vision techniques. Developed by Google, MediaPipe is a versatile framework renowned for its proficiency in detecting hand keypoints, surpassing OpenPose in terms of stability and accuracy. The approach for 3D detection of human hand keypoints involved the independent 2D detection of keypoints for each camera with MediaPipe, followed by the conversion of these 2D positions from both cameras into depth values using stereo-vision techniques.

Subsequent to, and based on the evaluation of MediaPipe's performance and the analysis of kinematic retargeting algorithms presented in Chapter 5, the BioIK algorithm was selected for mapping human movements to the robotic system composed of the robotic arm and hand. BioIK was favoured due to its ability to control the robotic arm and hand simultaneously.

Considering the kinematic retargeting algorithm, a similar implementation approach as described in Chapter 4 was employed, with additional BioIK goals introduced and adjustments made to the goals weights. The goals and their respective weights are detailed below:

- Position goals for fingertips mapping with weights of 1.0;
- Position goals for knuckles mapping with weights of 0.2;
- Position goals for wrist mapping with a weight of 0.25;
- Direction goals for proximal phalanges of the primary fingers with weights of 0.1;
- Direction goals for intermediate and distal phalanges of the thumb with weights of 0.1;
- Joint function goals for dealing with the coupled joints with a weight of 1.0;
- Centre joints goal with a weight of 0.1;
- Minimal displacement goal with a weight of 0.15;
- Joint function goal to ensure a maximum wrist extension with a weight of 0.5;
- Joint function goal to ensure a workspace above the base plane of UR5 with a weight of 0.15.

It should be noted that the penultimate goal mentioned aims to prevent collisions between the robotic hand's forearm and the bonnet where the paint quality inspection took place. Considering the physical properties of the Shadow Hand, if the hand and respective forearm were horizontally aligned, i.e. with null flexion of the wrist, the forearm would collide with the bonnet.

In summary, the procedure for performing this task involved capturing the keypoints of the human hand while performing the paint quality inspection and subsequently providing them to the BioIK algorithm, which aims to determine the positions of each joint in the robotic arm and hand, thereby replicating the previously recorded movement, including the intrinsic hand movements and its spatial position.

Once all the steps were well defined, experimental tests were conducted to evaluate the developed algorithm. Throughout the various iterations of tests carried out, slight differences (mostly less than a centimetre) were observed in the depth component, normal to the surface, of the acquired points. Consequently, in some instances, the robotic hand exhibited slight deviations from the surface. Nonetheless, despite these minor discrepancies, the algorithm proved effective in replicating human movements, as evidenced by the series of frames presented in Figure 6.14, extracted from a demonstration video showcasing the algorithm's execution. The respective video can be accessed through this [link](https://youtu.be/lpxIPF_6WYc)².



Figure 6.14: Automotive painting quality inspection process

An interesting observation derived from the conducted tests pertains to the execution time of the algorithm, which was measured as 200 milliseconds. Notably, despite BioIK being employed for joint control of both the robotic arm and hand in this specific case, the processing time decreased compared to when the algorithm solely encompassed control of the robotic hand, as evidenced in Section 5.2.3. Further investigation revealed that this discrepancy in execution time was attributed to the utilisation of OpenPose, an algorithm known for its computational intensity, which consequently impacted the available computational resources and led to an increased execution time for BioIK. However, when performing tests under equal circumstances, where OpenPose was not utilised, the execution times for both cases were equivalent (200 milliseconds), even in the presence of a higher number of joints requiring computation in this particular case.

Unfortunately, the extra objective of integrating a component within the system to actively assess the paint quality could not be accomplished due to the limitations imposed by the project and dissertation timeline.

²https://youtu.be/lpxIPF_6WYc

Chapter 7

Conclusions

The work developed in the context of this dissertation fits into the "CIBERTOQUE" project. It focuses on automating two specific tasks related to the automotive industry, specifically within Stellantis. These two tasks include wiring fitting and automotive painting quality inspection, with a specific emphasis on the replication of human movements.

The proposed solution involves utilising a robotic system comprising an anthropomorphic robotic hand, Shadow Dexterous Hand, coupled to a robotic arm, specifically the UR5, to predictably provide the necessary dexterity for accomplishing the aforementioned tasks. Notably, the research primarily focuses on controlling the robotic hand and optimising its performance to achieve precise manipulation during the designated tasks.

Considering the goal of controlling the Shadow Hand, three kinematic retargeting are analysed in detail: DexPilot, BioIK and TeachNET. These algorithms are specifically designed to replicate the positions obtained from human hands within the robotic hand.

Based on the analysis and evaluation of the presented algorithms, DexPilot and TeachNET proved to be more stable. Nevertheless, it was found that the instability observed with the BioIK algorithm resulted from the acquisition method used, OpenPose. Additionally, an experimental test involving 10 participants was conducted to assess the algorithms' performance. The results indicated that DexPilot exhibited the most favourable suitability for the wiring fitting task. However, the grasp position generated by this algorithm did not allow the clip insertion in its respective location.

Consequently, considering all the existing limitations, it was designed a clip adapter to ensure a larger grasp area for Shadow Hand and a novel algorithm was specifically designed. This new algorithm involved commanding the robotic hand with predefined postures carefully determined to optimise the hand's ability to firmly hold the clip. Applying this method resulted in the successful completion of the task at hand.

Considering the automotive painting quality inspection, thanks to the utilisation of a new method for capturing the human hand position, namely MediaPipe, which demonstrated enhanced stability compared to OpenPose, the BioIK algorithm was applied. Under this scenario, the BioIK algorithm exhibited improved performance, thus validating the earlier assertion that the limitation

resided in the acquisition of human hand data. This algorithm allowed a successful replication of movements resembling those performed by a human operator during the paint quality inspection task. Hence, the task can be deemed as partially accomplished since project time constraints did not allow for the accomplishment of the additional goal of effectively performing the quality inspection.

In conclusion, it can be affirmed that the primary objectives have been achieved, as the designated tasks within the project were successfully implemented and demonstrated within a relevant environment.

7.1 Future Work

In terms of prospects for enhancing the achievements of this dissertation, the following aspects are worth considering:

- Exploration of alternative methods for acquiring keypoints from the human hand to enhance accuracy and reliability. One viable option is the utilisation of a data acquisition glove, such as the Shadow Glove, which comes with associated costs but offers potential improvements in capturing hand movements;
- Incorporation of a sophisticated localisation system in order to allow the real-time correction of the clip's position and orientation before clipping;
- Addition of a sensory component, utilising the sensors incorporated in the Shadow Dexterous Hand fingertips, to the automotive paint quality inspection process to ensure uniform contact pressure throughout the whole inspection process;
- Development of a simulated environment resembling the real-world scenario, including the clips and their corresponding clipping positions and the bonnet, enabling the implementation of reinforcement learning algorithms in order to optimise the performance of the tasks;
- Integration of the effective inspection of automotive paint quality by incorporating DIGIT sensors.

Considering the application of the aforementioned points, it is undeniable that the proposed approach would experience a substantial improvement in terms of capabilities and overall effectiveness. Consequently, such advancements would propel the work to a significantly heightened level of achievement and impact.

7.2 Scientific Contributions

The main scientific contributions resulting from the research conducted throughout this dissertation were the following:

- Publication of the paper “Deep Convolutional Neural Networks applied to Hand Keypoints Estimation” in 2023 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC);
- Comparative analysis of the different kinematic retargeting algorithms utilising the real Shadow Hand;
- Public availability of the source code generated during this dissertation.

To conclude, mention that the publication of the mentioned article resulted in an invitation to submit an article to a scientific journal, namely "Journal of Intelligent & Robotic Systems".

Appendix A

Kinematic Retargeting Algorithms Test Results

Table A.1: Participant 1 score for each attempt utilizing TeachNET algorithm

TeachNET														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	2	2	4	4	4	4	0	4	4	0	4	4	4	0

Table A.2: Participant 1 score for each attempt utilizing BioIK algorithm

BioIK														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
2	0	4	2	2	4	0	4	1	4	0	0	1	1	4

Table A.3: Participant 1 score for each attempt utilizing DexPilot algorithm

DexPilot														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	0	2	4	1	4	4	4	4	4	4	2	4	2	0

Table A.4: Participant 2 score for each attempt utilizing TeachNET algorithm

TeachNET														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	4	4	4	0	4	4	4	4	4	4	4	2	4

Table A.5: Participant 2 score for each attempt utilizing BioIK algorithm

BioIK														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att.1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	0	0	4	4	4	1	2	1	4	4	0	4	1

Table A.6: Participant 2 score for each attempt utilizing DexPilot algorithm

DexPilot														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att.1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	4	4	4	4	4	2	4	4	4	4	4	4	4

Table A.13: Participant 5 score for each attempt utilizing TeachNET algorithm

TeachNET														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	0	4	4	4	2	4	4	4	4	0	4	4	4

Table A.14: Participant 5 score for each attempt utilizing BioIK algorithm

BioIK														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	1	4	4	4	4	4	4	2	4	0	0	4	4	1

Table A.15: Participant 5 score for each attempt utilizing DexPilot algorithm

DexPilot														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
0	4	4	4	4	4	4	4	1	4	4	4	4	4	4

Table A.19: Participant 7 score for each attempt utilizing TeachNET algorithm

TeachNET														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4

Table A.20: Participant 7 score for each attempt utilizing BioIK algorithm

BioIK														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att.1	Att. 2	Att. 3	Att. 4	Att. 5
4	1	4	1	1	4	4	2	4	0	1	0	0	1	2

Table A.21: Participant 7 score for each attempt utilizing DexPilot algorithm

DexPilot														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att.1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	4	4	4	4	4	4	1	4	4	1	4	4	4

Table A.28: Participant 10 score for each attempt utilizing TeachNET algorithm

TeachNET														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	2	4	4	4	4	4	4	4	4	4	4	4	1	1

Table A.29: Participant 10 score for each attempt utilizing BioIK algorithm

BioIK														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	1	4	4	1	4	4	4	1	2	2	4	0	4	4

Table A.30: Participant 10 score for each attempt utilizing DexPilot algorithm

DexPilot														
Clip 1					Clip 2					Clip 3				
Attempt 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5	Att. 1	Att. 2	Att. 3	Att. 4	Att. 5
4	4	4	4	4	4	2	4	4	4	2	4	4	4	4

References

- [1] Y. Zhao, *Current Status and Industrialization Development of Industrial Robot Technology*. 2021, vol. 81, pp. 804–808. DOI: [10.1007/978-3-030-79197-1_117](https://doi.org/10.1007/978-3-030-79197-1_117). [Online]. Available: https://doi.org/10.1007/978-3-030-79197-1_117.
- [2] H. N. Ghafil and K. Jármai, “Research and application of industrial robot manipulators in vehicle and automotive engineering, a survey,” K. Jármai and B. Bolló, Eds., pp. 611–623, 2018. DOI: [10.1007/978-3-319-75677-6_53](https://doi.org/10.1007/978-3-319-75677-6_53). [Online]. Available: https://doi.org/10.1007/978-3-319-75677-6_53.
- [3] M. Bartoš, V. Bulej, M. Bohušík, J. Stanček, V. Ivanov, and P. Macek, “An overview of robot applications in automotive industry,” *Transportation Research Procedia*, vol. 55, pp. 837–844, 2021, 14th International scientific conference on sustainable, modern and safe transport, ISSN: 2352-1465. DOI: [10.1016/j.trpro.2021.07.052](https://doi.org/10.1016/j.trpro.2021.07.052). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352146521004543>.
- [4] J. Persson, “Industrial robots in competitive automotive components manufacturing,” 1984, pp. 175–182. [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-0021726584%5C&partnerID=40&md5=9bf7149b846f9cf451bb0611b64d702d>.
- [5] C. Rosa, F. Silva, and L. P. Ferreira, “Improving the quality and productivity of steel wire-rope assembly lines for the automotive industry,” *Procedia Manufacturing*, vol. 11, pp. 1035–1042, 2017, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy, ISSN: 2351-9789. DOI: [10.1016/j.promfg.2017.07.214](https://doi.org/10.1016/j.promfg.2017.07.214). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2351978917304225>.
- [6] Shadow Robot Company, London, GBR, *Shadow dexterous hand technical specification*, Accessed: Nov. 6, 2022, 2021. [Online]. Available: https://www.shadowrobot.com/wp-content/uploads/2022/03/shadow_dexterous_hand_e_technical_specification.pdf.
- [7] Universal Robots, Odense, DNK, *Ur5 technical specifications*, Accessed: Dez. 14, 2022, 2016. [Online]. Available: https://www.universal-robots.com/media/50588/ur5_en.pdf.

- [8] P. M. Kebria, S. Al-wais, H. Abdi, and S. Nahavandi, “Kinematic and dynamic modelling of ur5 manipulator,” in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 004 229–004 234. DOI: [10.1109/SMC.2016.7844896](https://doi.org/10.1109/SMC.2016.7844896). [Online]. Available: <https://doi.org/10.1109/SMC.2016.7844896>.
- [9] M. Santello, G. Baud-Bovy, and H. Jörntell, “Neural bases of hand synergies,” *Frontiers in Computational Neuroscience*, no. MAR, 2013. DOI: [10.3389/fncom.2013.00023](https://doi.org/10.3389/fncom.2013.00023). [Online]. Available: <https://doi.org/10.3389/fncom.2013.00023>.
- [10] N. J. Jarque-Bou, A. Scano, M. Atzori, and H. Müller, “Kinematic synergies of hand grasps: A comprehensive study on a large publicly available dataset,” *Journal of NeuroEngineering and Rehabilitation*, vol. 16, no. 1, 2019. DOI: [10.1186/s12984-019-0536-6](https://doi.org/10.1186/s12984-019-0536-6). [Online]. Available: <https://doi.org/10.1186/s12984-019-0536-6>.
- [11] M. Atzori and H. Muller, “The ninapro database: A resource for semg naturally controlled robotic hand prosthetics,” vol. 2015–November, 2015, pp. 7151–7154. DOI: [10.1109/EMBC.2015.7320041](https://doi.org/10.1109/EMBC.2015.7320041). [Online]. Available: <https://doi.org/10.1109/EMBC.2015.7320041>.
- [12] H. Abdi and L. J. Williams, “Principal component analysis,” *WIREs Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010. DOI: [10.1002/wics.101](https://doi.org/10.1002/wics.101). [Online]. Available: <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.101>.
- [13] G. Salvietti, “Replicating human hand synergies onto robotic hands: A review on software and hardware strategies,” *Frontiers in Neurorobotics*, vol. 12, no. JUN, 2018. DOI: [10.3389/fnbot.2018.00027](https://doi.org/10.3389/fnbot.2018.00027). [Online]. Available: <https://doi.org/10.3389/fnbot.2018.00027>.
- [14] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” 2012, pp. 5026–5033. DOI: [10.1109/IROS.2012.6386109](https://doi.org/10.1109/IROS.2012.6386109). [Online]. Available: <https://doi.org/10.1109/IROS.2012.6386109>.
- [15] F. M. Ribeiro, “Modeling and realistic simulation of a dexterous robotic manipulator,” M.S. thesis, Faculdade de Engenharia da Universidade do Porto, Oct. 2022.
- [16] M. Lambeta, P.-W. Chou, S. Tian, *et al.*, “DIGIT: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3838–3845, Jul. 2020. DOI: [10.1109/lra.2020.2977257](https://doi.org/10.1109/lra.2020.2977257). [Online]. Available: <https://doi.org/10.1109/lra.2020.2977257>.
- [17] M. Atzori, A. Gijsberts, S. Heynen, *et al.*, “Experiences in the creation of an electromyography database to help hand amputated persons,” *Studies in Health Technology and Informatics*, vol. 180, pp. 828–832, 2012. DOI: [10.3233/978-1-61499-101-4-828](https://doi.org/10.3233/978-1-61499-101-4-828). [Online]. Available: <https://doi.org/10.3233/978-1-61499-101-4-828>.

- [18] M. Atzori, A. Gijssberts, S. Heynen, *et al.*, “Building the ninapro database: A resource for the biorobotics community,” in *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012, pp. 1258–1265. DOI: [10.1109/BioRob.2012.6290287](https://doi.org/10.1109/BioRob.2012.6290287). [Online]. Available: <https://doi.org/10.1109/BioRob.2012.6290287>.
- [19] N. J. Jarque-Bou, M. Atzori, and H. Müller, “A large calibrated database of hand movements and grasps kinematics,” *Scientific Data*, vol. 7, no. 1, 2020. DOI: [10.1038/s41597-019-0349-2](https://doi.org/10.1038/s41597-019-0349-2). [Online]. Available: <https://doi.org/10.1038/s41597-019-0349-2>.
- [20] T. Cerqueira, F. M. Ribeiro, V. H. Pinto, J. Lima, and G. Gonçalves, “Glove prototype for feature extraction applied to learning by demonstration purposes,” *Applied Sciences (Switzerland)*, vol. 12, no. 21, 2022. DOI: [10.3390/app122110752](https://doi.org/10.3390/app122110752). [Online]. Available: <https://doi.org/10.3390/app122110752>.
- [21] Z. Guo, W. Zeng, T. Yu, *et al.*, “Vision-based finger tapping test in patients with parkinson’s disease via spatial-temporal 3d hand pose estimation,” *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 8, pp. 3848–3859, 2022. DOI: [10.1109/JBHI.2022.3162386](https://doi.org/10.1109/JBHI.2022.3162386). [Online]. Available: <https://doi.org/10.1109/JBHI.2022.3162386>.
- [22] F. Xiong, B. Zhang, Y. Xiao, *et al.*, “A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 793–802. DOI: [10.1109/ICCV.2019.00088](https://doi.org/10.1109/ICCV.2019.00088). [Online]. Available: <https://doi.org/10.1109/ICCV.2019.00088>.
- [23] J. Redmon and A. Farhadi, *Yolov3: An incremental improvement*, 2018. DOI: [10.48550/ARXIV.1804.02767](https://arxiv.org/abs/1804.02767). [Online]. Available: <https://arxiv.org/abs/1804.02767>.
- [24] S. Yuan, Q. Ye, G. Garcia-Hernando, and T.-K. Kim, *The 2017 hands in the million challenge on 3d hand pose estimation*, 2017. DOI: [10.48550/ARXIV.1707.02237](https://arxiv.org/abs/1707.02237). [Online]. Available: <https://arxiv.org/abs/1707.02237>.
- [25] R. Shamalik and S. Koli, “Deephands: Dynamic hand gesture detection with depth estimation and 3d reconstruction from monocular rgb data,” *Sadhana - Academy Proceedings in Engineering Sciences*, vol. 47, no. 4, 2022. DOI: [10.1007/s12046-022-02026-7](https://doi.org/10.1007/s12046-022-02026-7). [Online]. Available: <https://doi.org/10.1007/s12046-022-02026-7>.
- [26] N. Zhang, Y. Zhao, G. Gu, and X. Zhu, “Synergistic control of soft robotic hands for human-like grasp postures,” *Science China Technological Sciences*, vol. 65, no. 3, pp. 553–568, 2022. DOI: [10.1007/s11431-021-1944-y](https://doi.org/10.1007/s11431-021-1944-y). [Online]. Available: <https://doi.org/10.1007/s11431-021-1944-y>.

- [27] A. A. Portnova-Fahreeva, F. Rizzoglio, I. Nisky, M. Casadio, F. A. Mussa-Ivaldi, and E. Rombokas, “Linear and non-linear dimensionality-reduction techniques on full hand kinematics,” *Frontiers in Bioengineering and Biotechnology*, vol. 8, 2020. DOI: [10.3389/fbioe.2020.00429](https://doi.org/10.3389/fbioe.2020.00429). [Online]. Available: <https://doi.org/10.3389/fbioe.2020.00429>.
- [28] E. Battaglia, M. Kasman, and A. M. Fey, “Moving past principal component analysis: Non-linear dimensionality reduction towards better hand pose synthesis,” in *2022 International Symposium on Medical Robotics (ISMR)*, 2022, pp. 1–7. DOI: [10.1109/ISMR48347.2022.9807580](https://doi.org/10.1109/ISMR48347.2022.9807580). [Online]. Available: <https://doi.org/10.1109/ISMR48347.2022.9807580>.
- [29] B.-j. Kim, “Active visual learning and recognition using incremental kernel pca,” S. Zhang and R. Jarvis, Eds., pp. 585–592, 2005. DOI: [10.1007/11589990_61](https://doi.org/10.1007/11589990_61). [Online]. Available: https://doi.org/10.1007/11589990_61.
- [30] T.-J. Chin and D. Suter, “Incremental kernel principal component analysis,” *IEEE Transactions on Image Processing*, vol. 16, no. 6, pp. 1662–1674, 2007. DOI: [10.1109/TIP.2007.896668](https://doi.org/10.1109/TIP.2007.896668). [Online]. Available: <https://doi.org/10.1109/TIP.2007.896668>.
- [31] O. Glauser, S. Wu, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung, “Interactive hand pose estimation using a stretch-sensing soft glove,” *ACM Trans. Graph.*, vol. 38, no. 4, Jul. 2019, ISSN: 0730-0301. DOI: [10.1145/3306346.3322957](https://doi.org/10.1145/3306346.3322957). [Online]. Available: <https://doi.org/10.1145/3306346.3322957>.
- [32] A. Handa, K. Van Wyk, W. Yang, *et al.*, “Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system,” All Open Access, Green Open Access, 2020, pp. 9164–9170. DOI: [10.1109/ICRA40945.2020.9197124](https://doi.org/10.1109/ICRA40945.2020.9197124). [Online]. Available: <https://doi.org/10.1109/ICRA40945.2020.9197124>.
- [33] S. Starke, N. Hendrich, S. Magg, and J. Zhang, “An efficient hybridization of genetic algorithms and particle swarm optimization for inverse kinematics,” 2016, pp. 1782–1789. DOI: [10.1109/ROBIO.2016.7866587](https://doi.org/10.1109/ROBIO.2016.7866587). [Online]. Available: <https://doi.org/10.1109/ROBIO.2016.7866587>.
- [34] P. Ruppel, N. Hendrich, S. Starke, and J. Zhang, “Cost functions to specify full-body motion and multi-goal manipulation tasks,” 2018, pp. 3152–3159. DOI: [10.1109/ICRA.2018.8460799](https://doi.org/10.1109/ICRA.2018.8460799). [Online]. Available: <https://doi.org/10.1109/ICRA.2018.8460799>.
- [35] A. Fabisch, M. Uliano, D. Marschner, M. Laux, J. Brust, and M. Controzzi, “A modular approach to the embodiment of hand motions from human demonstrations,” All Open Access, Green Open Access, vol. 2022–November, 2022, pp. 801–808. DOI: [10.1109/Humanoids53995.2022.10000165](https://doi.org/10.1109/Humanoids53995.2022.10000165). [Online]. Available: <https://doi.org/10.1109/Humanoids53995.2022.10000165>.

- [36] J. Romero, D. Tzionas, and M. J. Black, “Embodied hands: Modeling and capturing hands and bodies together,” *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 245:1–245:17, vol. 36, no. 6, Nov. 2017.
- [37] S. Li, X. Ma, H. Liang, *et al.*, “Vision-based teleoperation of shadow dexterous hand using end-to-end deep neural network,” All Open Access, Green Open Access, vol. 2019-May, 2019, pp. 416–422. DOI: [10.1109/ICRA.2019.8794277](https://doi.org/10.1109/ICRA.2019.8794277). [Online]. Available: <https://doi.org/10.1109/ICRA.2019.8794277>.
- [38] S. Yuan, Q. Ye, B. Stenger, S. Jain, and T.-K. Kim, “Bighand2.2m benchmark: Hand pose dataset and state of the art analysis,” All Open Access, Green Open Access, vol. 2017-January, 2017, pp. 2605–2613. DOI: [10.1109/CVPR.2017.279](https://doi.org/10.1109/CVPR.2017.279). [Online]. Available: <https://doi.org/10.1109/CVPR.2017.279>.
- [39] M. Oberweger and V. Lepetit, “Deepprior++: Improving fast and accurate 3d hand pose estimation,” All Open Access, Green Open Access, vol. 2018-January, 2017, pp. 585–594. DOI: [10.1109/ICCVW.2017.75](https://doi.org/10.1109/ICCVW.2017.75). [Online]. Available: <https://doi.org/10.1109/ICCVW.2017.75>.
- [40] Y. Lin, S. Ren, M. Clevenger, and Y. Sun, “Learning grasping force from demonstration,” in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 1526–1531. DOI: [10.1109/ICRA.2012.6225222](https://doi.org/10.1109/ICRA.2012.6225222). [Online]. Available: <https://doi.org/10.1109/ICRA.2012.6225222>.
- [41] N. Pham and T. Yoshimi, “Adaptive learning of hand movement in human demonstration for robot action,” English, *Journal of Robotics and Mechatronics*, vol. 29, no. 5, pp. 919–927, Oct. 2017, ISSN: 0915-3942. DOI: [10.20965/jrm.2017.p0919](https://doi.org/10.20965/jrm.2017.p0919). [Online]. Available: <https://doi.org/10.20965/jrm.2017.p0919>.
- [42] G. Solak and L. Jamone, “Learning by demonstration and robust control of dexterous in-hand robotic manipulation skills,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 8246–8251. DOI: [10.1109/IROS40897.2019.8967567](https://doi.org/10.1109/IROS40897.2019.8967567). [Online]. Available: <https://doi.org/10.1109/IROS40897.2019.8967567>.
- [43] C. Mueller, J. Venicx, and B. Hayes, “Robust robot learning from demonstration and skill repair using conceptual constraints,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6029–6036. DOI: [10.1109/IROS.2018.8594133](https://doi.org/10.1109/IROS.2018.8594133). [Online]. Available: <https://doi.org/10.1109/IROS.2018.8594133>.
- [44] P. Ruppel and J. Zhang, “Learning object manipulation with dexterous hand-arm systems from human demonstration,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5417–5424. DOI: [10.1109/IROS45743.2020.9340966](https://doi.org/10.1109/IROS45743.2020.9340966). [Online]. Available: <https://doi.org/10.1109/IROS45743.2020.9340966>.

- [45] O. M. Andrychowicz, B. Baker, M. Chociej, *et al.*, “Learning dexterous in-hand manipulation,” *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020. DOI: [10.1177/0278364919887447](https://doi.org/10.1177/0278364919887447). [Online]. Available: <https://doi.org/10.1177/0278364919887447>.
- [46] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033. DOI: [10.1109/IROS.2012.6386109](https://doi.org/10.1109/IROS.2012.6386109). [Online]. Available: <https://doi.org/10.1109/IROS.2012.6386109>.
- [47] F. Ellrich, M. Bauer, N. Schreiner, *et al.*, “Terahertz quality inspection for automotive and aviation industries,” *Journal of Infrared, Millimeter, and Terahertz Waves*, vol. 41, no. 4, pp. 470–489, 2020. DOI: [10.1007/s10762-019-00639-4](https://doi.org/10.1007/s10762-019-00639-4). [Online]. Available: <http://doi.org/10.1007/s10762-019-00639-4>.
- [48] W.-L. Mao, Y.-Y. Chiu, B.-H. Lin, *et al.*, “Integration of deep learning network and robot arm system for rim defect inspection application,” *Sensors*, vol. 22, no. 10, 2022. DOI: [10.3390/s22103927](https://doi.org/10.3390/s22103927). [Online]. Available: <https://doi.org/10.3390/s22103927>.
- [49] B. M. Santos, P. Pais, F. M. Ribeiro, J. Lima, G. Gonçalves, and V. H. Pinto, “Deep convolutional neural networks applied to hand keypoints estimation,” in *2023 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2023, pp. 93–98. DOI: [10.1109/ICARSC58346.2023.10129621](https://doi.org/10.1109/ICARSC58346.2023.10129621). [Online]. Available: <https://doi.org/10.1109/ICARSC58346.2023.10129621>.
- [50] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “Openpose: Realtime multi-person 2d pose estimation using part affinity fields,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021, All Open Access, Green Open Access. DOI: [10.1109/TPAMI.2019.2929257](https://doi.org/10.1109/TPAMI.2019.2929257). [Online]. Available: <https://doi.org/10.1109/TPAMI.2019.2929257>.
- [51] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, “2d human pose estimation: New benchmark and state of the art analysis,” 2014, pp. 3686–3693. DOI: [10.1109/CVPR.2014.471](https://doi.org/10.1109/CVPR.2014.471). [Online]. Available: <https://doi.org/10.1109/CVPR.2014.471>.
- [52] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, “Microsoft coco: Common objects in context,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, 2014, All Open Access, Bronze Open Access, Green Open Access. DOI: [10.1007/978-3-319-10602-1_48](https://doi.org/10.1007/978-3-319-10602-1_48). [Online]. Available: https://doi.org/10.1007/978-3-319-10602-1_48.