Titre: Title:	Development of a Versatile Exploration and Optimization Tool for the Conceptual Design of Serial Robot Manipulators
Auteur: Author:	Adrien Cholet
Date:	2025
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Cholet, A. (2025). Development of a Versatile Exploration and Optimization Tool for the Conceptual Design of Serial Robot Manipulators [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/66945/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/66945/
Directeurs de recherche: Advisors:	Sofiane Achiche
Programme: Program:	Génie mécanique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Development of a Versatile Exploration and Optimization Tool for the Conceptual Design of Serial Robot Manipulators

ADRIEN CHOLET
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie mécanique

Juillet 2025

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Development of a Versatile Exploration and Optimization Tool for the Conceptual Design of Serial Robot Manipulators

présenté par **Adrien CHOLET**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Luc BARON, président
Sofiane ACHICHE, membre et directeur de recherche
Gabriel PICARD-KRASHEVSKI, membre

ACKNOWLEDGEMENTS

First and foremost, I would like to express my most sincere gratitude to my research director, Professor Sofiane Achiche. He was the one to initially motivate me to engage in a research-based master's degree. I thank him for his kindness and his continuous support throughout the project as well as his constructive and critical feedback during the thesis writing process.

I would also like to thank the industrial partner of the project, KINOVA Robotics, and particularly Jean-François Gamache, PhD, who guided and advised me during the length of the project. I appreciate his kindness and his wisdom, which prevented me from losing track of the objectives in this very large scale project.

I also thank the MITACS Program, the Haut-de-France region and the UTC foundation for the financial support which allowed me to spend these two years working on this project.

I would like to extend my gratitude to my friends from the CoSIM Lab for the good spirits and the daily UNO games that either kept us sane, or drove us mad, depending on the day.

Finally, I would like to thank my friends and family for the unwavering love and support which have been a pillar throughout the course of my studies.

Note : I would like to acknowledge the assistance of ChatGPT for formulation and syntax refinement when needed. Nevertheless, I take full ownership of the intellectual contributions, analyses and conclusions.

RÉSUMÉ

Les manipulateurs robotiques sont au cœur d'un large éventail d'applications, allant des tâches de soudage industriel aux interventions chirurgicales de précision. Face à la demande croissante d'applications innovantes, les entreprises de robotique doivent adapter leur processus de conception afin d'évaluer efficacement les nouveaux concepts et les compromis en matière de performances. Constatant le manque d'efficacité des premières étapes de la conception, cette thèse présente le développement d'un outil polyvalent conçu pour optimiser et explorer la conception préliminaire des manipulateurs robotiques à chaîne cinématique ouverte. Une analyse approfondie de la littérature a révélé les principales limites des méthodologies existantes, notamment en ce qui concerne la généralisabilité des modèles, la polyvalence des applications possibles et la flexibilité algorithmique. Cette étude vise donc à relever ces défis en introduisant un outil complet intégrant à la fois des approches de conception directe et inverse, permettant une évaluation efficace de concepts de manipulateurs robotiques.

La méthodologie développée dans ce mémoire de maîtrise utilise un modèle de manipulateur intégrant des composants cinématiques, dynamiques, structurels et d'actionnement. Le modèle cinématique utilise la convention des séquences de transformation élémentaires pour l'interprétabilité, tandis que les modèles dynamiques et structurels proposent des représentations simplifiées des propriétés des membrures. Les actionneurs, représentés par un moteur et un réducteur, sont utilisés pour dériver trois modèles de limites de performance à des fins d'évaluation différentes. Un module de cinématique inverse optimisé est également inclus, guidant la sélection de la configuration des joints avec l'optimisation d'un objectif secondaire.

L'outil comprend un ensemble de fonctions d'évaluation de critères de performances développées suite à une études des indicateurs fréquemment mentionnés dans la littérature et les fiches techniques commerciales. Ces fonctions englobent des critères d'évaluation globaux, ainsi que des indicateurs spécifiques à des cas d'utilisation, définis par des positions dans l'espace ou des chemins ou trajectoires à suivre. Une nouvelle application des méthodes d'optimisation est proposée pour dériver des indices de performance locaux globalisés en identifiant les configurations articulaires critiques. L'efficacité des fonctions d'évaluation est démontrée à travers l'analyse d'un modèle du robot collaboratif Kinova GEN3, les évaluations nécessitant généralement moins de deux minutes.

Le module de conception inverse intègre un ensemble de variables de conception, notamment des paramètres cinématiques et des configurations d'actionneurs, ce qui lui permet de s'adapter à diverses applications robotiques. Une sélection d'algorithmes d'optimisation

largement utilisés dans la littérature est incluse afin de tirer parti de leurs atouts respectifs pour différents problèmes de conception. La flexibilité de l'outil permet à l'utilisateur d'adapter les problèmes d'optimisation en utilisant des critères de performance globaux ou spécifiques à une tâche, surmontant ainsi les limites traditionnelles observées dans la littérature.

Afin de déterminer les stratégies d'optimisation les plus efficaces, l'étude introduit un système de classification des problèmes de conception inverse. Six classes sont définies, variant en termes de dimensionnalité, de contraintes et de types de variables. Une analyse comparative des performances des algorithmes dans ces différents cas révèle que différents algorithmes excellent dans différents scénarios. Par exemple, l'algorithme CMA-ES est généralement performant pour tous les problèmes à objectif unique, trouvant la meilleure solution dans 67 % des problèmes étudiés, tandis que l'algorithme d'optimisation bayésienne identifie les meilleurs résultats avec 20 à 30 % d'évaluations en moins pour les problèmes à faible dimension. En ce qui concerne les problèmes à objectifs multiples, l'optimisation bayésienne excelle dans les scénarios à faible dimension, tandis que l'algorithme NSGA-II est plus performant dans les dimensions supérieures. Un arbre de décision est présenté dans le but d'aider l'utilisateur dans le choix de l'algorithme en fonction des caractéristiques du problème.

Dans l'ensemble, ce mémoire contribue au domaine de l'optimisation des manipulateurs en développant un outil qui répond efficacement à un large éventail de défis de conception et fournit une base pour de futures applications académiques et industrielles. La mise en œuvre de la méthodologie sous la forme d'un outil Python pour KINOVA Robotics représente une avancée pratique vers l'accélération du processus de conception et la promotion de l'innovation dans les concepts de manipulateurs robotiques.

ABSTRACT

Robotic manipulators are at the heart of a wide range of applications, ranging from industrial welding tasks, to precise medical surgeries. As the demand increases for innovative applications, robotics companies must adapt the design process to efficiently evaluate novel concepts and performance trade-offs. In observation of the lack of efficiency in the early stages of design, this thesis presents the development of a versatile framework designed to optimize and explore the conceptual design of serial robot manipulators. A thorough literature review revealed key limitations in existing methodologies, particularly concerning model generalizability, versatility in possible applications, and algorithmic flexibility. Thus, this study aims to address these challenges by introducing a comprehensive tool that integrates both direct and inverse design approaches, enabling the efficient evaluation of manipulator concepts.

The framework proposed in this master's thesis uses a manipulator model incorporating kinematic, dynamic, structural, and actuator components. The kinematic model uses Elementary Transform Sequences for interpretability, while the dynamic and structural models offer simplified representations of link properties. The actuators are represented with their motor and reducer components and three performance limit models are derived for different evaluation purposes. An optimized inverse kinematics module is also included, guiding configuration selection with the optimization of a secondary objective.

The framework includes a set of performance evaluation functions based on the literature and commercial data-sheets. These functions encompass criteria for both global and task-specific performance evaluation over position, path, and trajectory-based tasks. A novel application of optimization methods is proposed to derive globalized local performance indices by identifying critical joint configurations. The efficiency of the evaluation functions is demonstrated on a model of the Kinova GEN3 collaborative robot, with evaluations typically requiring less than two minutes.

The inverse design framework incorporates an extensive array of design variables, including kinematic parameters and actuator configurations, allowing it to accommodate several robotic applications. A selection of widely used optimization algorithms from the literature is implemented to leverage their respective strengths across different design problems. The tool's flexibility allows engineers to tailor optimization problems using either global or task-specific performance criteria, overcoming traditional limitations seen in other frameworks.

To determine the most effective optimization strategies, the study introduces a classification

scheme for inverse design problems. Six classes are defined, varying in dimensionality, constraints, and variable types. A comparative analysis of algorithm performance across these cases reveals that different algorithms excel in different scenarios. For instance, CMA-ES generally performs well across all mono-objective problems, finding the best design in 67% of the studied problems although the Bayesian optimization algorithm finds the best results in 20-30% less function evaluations in low-dimensional problems. In multi-objective contexts, Bayesian optimization outperforms the others in low-dimensional scenarios, whereas NSGA-II leads in higher dimensions. A decision tree is introduced with the aim to guide the user in algorithm selection depending on a problem's characteristics.

Overall, this research contributes to the field of manipulator optimization by developing a comprehensive framework that effectively addresses a diverse range of design challenges and provides a foundation for future academic and industrial applications. The implementation of the framework as a Python-based tool for KINOVA Robotics signifies a practical advancement towards streamlining the design process and fostering innovation in robotic manipulator concepts.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACO	Ant Colony Optimization
BO	Bayesian Optimization
CMA-ES	Covariance Matrix Adaptation - Evolution Strategy
DE	Differential Evolution
DH	Denavit-Hartenberg
DoF	Degree of freedom
DP	Design Parameter
DV	Design Variable
ETS	Elementary Transform Sequence
GA	Genetic Algorithm
GCI	Global Conditioning Index
HTM	Homogeneous Transformation Matrix
IK	Inverse Kinematics
LHS	Latin Hypercube Sampling
LM	Levenberg-Marquardt
MACO	Multi-Objective Ant Colony Optimization
MOEAD	Multi-Objective Evolutionary Algorithm with Decomposition
MOO	Multi-Objective Optimization
NSGA-II	Non-Dominated Sorted Genetic Algorithm II
NSPSO	Non-Dominated Sorting Particle Swarm Optimization
PSO	Particle Swarm Optimization
RFP	Request for Proposal
RNE	Recursive Newton-Euler
SLI	Structural Length Index
SQP	Sequential Quadratic Programming
TOPP	Time Optimal Path Parametrization

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

In a hospital across the world, a surgeon is sitting at a console in a cutting-edge operating room, performing a life-saving cardiac procedure. Meanwhile, in a quiet home, a man who lost mobility due to a spinal cord injury reaches for his glass of water and effortlessly eats his meal. Thousands of kilometers above his head, a drifting satellite is secured and repaired, preventing it from becoming dangerous space debris. All these situations may seem unrelated, but they share a common foundation, they are all enabled by the use of robot manipulators, highly versatile mechatronics systems designed to extend human capabilities in a variety of applications. The surgeon uses a surgical robot like the Da Vinci Surgical System [1] or Medtronic Hugo™ RAS system [2] to perform minimally invasive procedures with high precision, the man with mobility impairment has his wheelchair equipped with an assistive robot like the JACO arm from Kinova Robotics [3] and the drifting satellite is caught by an exploration robot arm similar to the Canadarm [4].

These examples are only a snapshot of the vast range of implementations of robotic manipulators. Historically, they were designed to enhance efficiency, precision and repeatability in industrial settings (assembly, welding, painting, etc.) [5]. In the 21th century, the uses spread to the medical field, education and research [6], remote exploration [7, 8], hazardous material handling [9], mining [10, 11], and more. The growing reliance on robotic systems reflects in global trends, with the International Federation of Robotics reporting a 10% increase in the number of industrial robots deployed in factories worldwide in 2024 [12]. To keep up with the expanding demands and applications, robotics companies must accelerate the design process and explore a wider range of innovative solutions.

However, the design of serial robot manipulators to meet performance requirements is a complex and multidisciplinary challenge that involves balancing multiple conflicting factors, such as payload capacity, workspace dexterity, energy consumption, precision, cost and safety. [13]. For companies involved in the design of robotic systems, the ability to efficiently generate and evaluate robot designs is crucial, particularly when responding to a Request for Proposal (RFP). An RFP response often involves performing feasibility studies to assess whether the company's expertise and resources are sufficient to design a system that meets a client's performance expectations while adhering to constraints such as time, budget, and industry standards.

Traditional design processes often involve iterative manual tuning, experience-based design choices, and extensive prototyping [13]. They also require the simultaneous contribution

of multiple engineers from different disciplines (mechanics, systems, electronics, control...), making the early design stages particularly inefficient in robotics design [14]. These constraints limit the ability to efficiently explore the vast design space of a robot manipulator and generate a competitive proposal in a short time-frame.

To remain competitive, robotics companies need automated tools that can accelerate the conceptual design phase to establish the resources needed to meet the objective performance requirements.

While robotic manipulator design has been widely studied, the current literature for conceptual design optimization is often limited to a narrow set of objectives, predefined use cases, and selective design variables, restricting their adaptability to diverse application needs. This gap makes it challenging to efficiently assess the feasibility of a manipulator design that balances performance and constraints for novel applications and industries.

To address this, this project aims to develop a versatile optimization and exploration tool for the conceptual design of serial robotic manipulators. By integrating a broad set of performance metrics and design parameters, this tool aims to enable rapid evaluation and optimization of robot configurations for diverse applications, improving early-stage design efficiency.

This thesis details the complete development of the proposed tool. Chapter 2 reviews the theoretical foundations and existing methodologies. Chapter 3 presents the modeling and optimization framework and its implementation. Chapter 4 demonstrates its capabilities through case studies. Finally, Chapter 5 discusses the obtained findings and limitations and Chapter 6 presents a conclusion and recommendations for future research directions.

CHAPTER 2 LITERATURE REVIEW AND THEORETICAL FOUNDATIONS

This chapter aims to present the theoretical foundations essential for the thesis, along with a review of the current literature on the subject. The first section introduces fundamental definitions and core concepts in robotics that are crucial to the development of the subsequent chapters. The second section describes relevant performance indicators to evaluate robot designs. The third section introduces key concepts in design optimization and critically reviews the existing literature on the optimization of serial robot manipulators. The fourth section discusses the limitations of the reviewed literature and the final section presents the rationale and objectives of the thesis.

2.1 Serial Robotic Manipulators

2.1.1 General Definitions and Scope of the Thesis

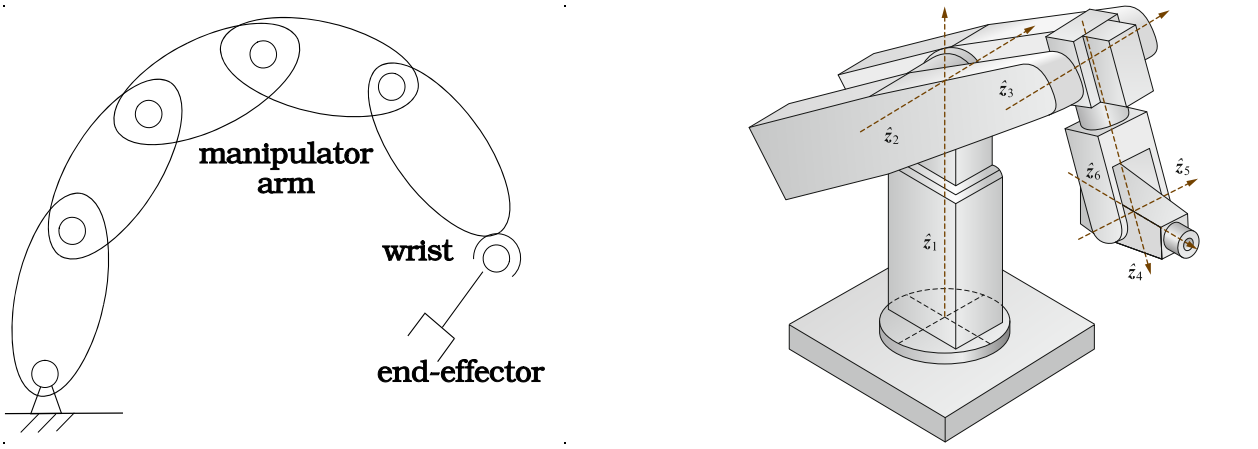
The term "robot" has many different definitions according to different sources and fields. In the context of this thesis, the simple technological definition presented by the ISO8373 norm will be considered, a "programmed actuated mechanism with a degree of autonomy to perform locomotion, manipulation or positioning" [15].

This definition is very large and groups various types of robots: manipulators, mobile robots, humanoid robots, soft robots and more. The category of interest for this project is robotic manipulators, mechanically articulated mechanisms controlled by actuators capable of moving and positioning an end-effector in a controlled manner.

The end effector of a manipulator is the tool, attached to a link, that interacts with the external environment. Its form varies based on the robot's intended function, such as a gripper for handling objects, a welding torch for fabrication, or a surgical instrument for medical applications.

More specifically, the focus is set on serial manipulators, often referred to as robotic arms, composed of a sequence of rigid links connected by joints in an open-chain configuration. Unlike parallel manipulators, which feature closed-chain architectures that enhance structural rigidity and precision at the cost of a more complex workspace and control, serial manipulators offer greater flexibility and reach, making them well-suited for a wide range of applications.

Within the category of serial robot manipulators, there are many different types of robots that are often classified based on their topology, the relative placement of the joints to one another and the type of the joints (prismatic, revolute, spherical...). While the methodology presented in this thesis could be extended to other joint types, the focus of the work is limited to revolute joints. These joints allow a 1-Degree of Freedom (DoF) relative motion between two adjacent links characterized by a rotation around an axis. Example representations of serial manipulators composed of revolute joints are illustrated in Figure 2.1.



(a) Schematic illustration of a serial manipulator. [16]

(b) Example of a 6-DoF serial manipulator. [17]

Figure 2.1 Schematic representation and example of a serial robot manipulator composed of revolute joints.

The design of serial robot manipulators follows a structured engineering process that ensures the system meets functional, performance, and operational requirements. As serial manipulators are complex mechatronic systems integrating mechanical, electrical, and control components, their development necessitates a multidisciplinary approach. The design process is generally divided into several stages, each addressing specific challenges and decisions. The focus of this thesis is set on the conceptual design phase.

The conceptual design phase is the foundation of the design process, where fundamental decisions about the manipulator's structure and capabilities are made. At this stage, engineers explore a wide range of configurations, considering factors such as degrees of freedom, workspace, kinematic structure, and actuator placement. The goal is to identify a concept that best satisfies the required task specifications, balancing criteria like payload capacity, reach and manipulability. Optimization plays a critical role in this phase, as selecting the right design early on can significantly impact performance and feasibility in later stages. The

output of this phase is a well-defined design concept that serves as a blueprint for subsequent development.

After the conceptual design phase, the preliminary design phase refines the selected concept by defining system architecture, major components, and critical parameters. The detailed design phase then translates this refined concept into precise manufacturing specifications, including component dimensions, material selection, and structural analyses. Finally, the production and prototyping phase brings the design to reality, using prototype testing to validate assumptions and refine the final product before full-scale production. The stages of the design process are illustrated in Figure 2.2.

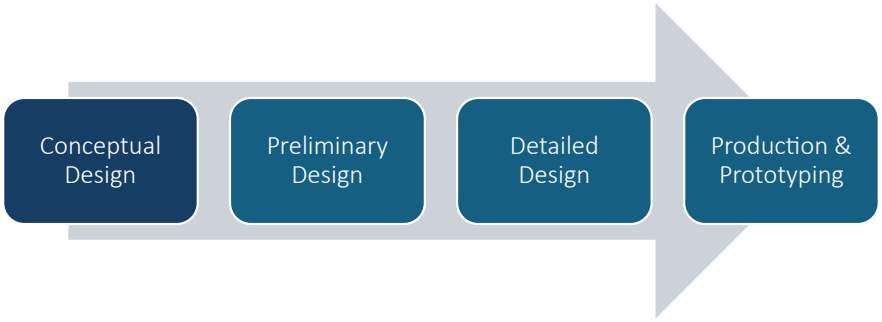


Figure 2.2 Main stages of the product design process

2.1.2 Introduction to Core Concepts of Robotics

Robotics is a large field with complex concepts and several books have been published to provide a complete overview of the field and its technical notions [16–18]. The aim of this section is to review the most important concepts as theoretical foundation for the next chapters.

Kinematics

A fundamental aspect of robotic manipulators is their ability to move and position their end effector accurately, which is governed by kinematics, the study of motion without consideration of the effects of involved forces. The core concept of kinematics is the conversion of the motion of a robot in the joint space (angular rotations of each joint) to a motion in the cartesian space (position and orientation of the end effector with respect to a reference frame). The configuration of the robot in the joint space is defined by a n-dimensional vector $\mathbf{q}$ composed of the angular position of each joint. In the cartesian space, the position and

orientation of the end effector is represented by its pose, a 6-dimensional vector $\mathbf{p}$ where the first three components indicate the position and the last three define the orientation.

The relationship between the joint space and the cartesian space of the robot is separated in two problems. Forward kinematics have the objective of finding the position and orientation of the end-effector considering the angular position of each joint. Inversely, inverse kinematics (IK) aims to find the angular position of each joint for the end-effector to reach a cartesian pose.

Forward kinematics can be computed using a Homogeneous Transformation Matrix (HTM) dependent on the angular position of each joint ${}^{\mathbf{base}}\mathbf{T}_{\mathbf{end-effector}}(\mathbf{q})$. The transformation matrix is a 4x4 matrix containing a 3x3 rotation matrix and the translation vector, allowing for the extraction of the Cartesian pose coordinates with respect to a reference frame. The representation of orientation depends on the chosen convention, such as Euler angles, fixed angles, or quaternions. Various kinematic representations exist for modeling these transformations. The Denavit-Hartenberg (DH) [19] convention provides a compact parameterization of joint transformations, while the Product of Exponentials [20] method and screw theory use screw motions and exponentials to describe link interactions. Alternatively, the Elementary Transform Sequence (ETS) method combines elementary transformations in a sequence, providing an intuitive way to model kinematic chains [21]. The complete set of end-effector poses achievable by varying the joint positions defines the robot's workspace.

Inverse kinematics requires to solve a system of nonlinear equations. Depending on the pose and the number of joints of the robot, there can be an infinite number of solutions, a finite set of solutions or no solution at all. For specific manipulator structures, the solution to the IK problem can be obtained analytically [17]. In general cases, the solution to an IK problem can be solved using numerical methods [22–25]. The numerical method used for inverse kinematics in this project is the damped least squares method, also known as Levenberg-Marquardt (LM) algorithm [26]. It was chosen because of its good performance with redundant robots (robots with more than six DoF) and its efficiency compared to other methods [27].

While forward and inverse kinematics describe the relationship between joint positions and end-effector poses, instantaneous kinematics focus on the relationship between joint velocities and end-effector velocities, crucial information for motion control and trajectory planning. The robot Jacobian $\mathbf{J}$, is a $6 \times n$ fundamental matrix that relates the velocity in joint space to the velocity in Cartesian space. It is derived by differentiating the forward kinematics equations with respect to time (Eq. 2.1).

$$\dot{\mathbf{p}} = \mathbf{J}\dot{\mathbf{q}} \tag{2.1}$$

The first three rows of the Jacobian map the joint velocities to the linear velocity of the end effector while the bottom three map the joint velocities to its angular velocity. The rank of the Jacobian is also the main tool used to analyze the singularities of the robot, configurations where the motion loses a degree of freedom in cartesian space. When the Jacobian loses rank, the robot is in a singular configuration.

Dynamics

While kinematics describes the motion of a robotic manipulator in terms of position, velocity, and acceleration without considering the forces that generate it, dynamics focuses on the relationship between forces, torques, and motion. A comprehensive understanding of a robot's behavior requires dynamic modeling, which is fundamental for control, simulation, and trajectory optimization.

The rigid-body dynamics of an n-DoF serial manipulator are typically described by the following equation of motion in joint space:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau} \tag{2.2}$$

Where $\mathbf{M}(\mathbf{q})$ is the inertia matrix (or mass matrix) representing the resistance of the robot to acceleration in a configuration $\mathbf{q}$. $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is the Coriolis and centrifugal matrix which accounts for forces due to joint velocities. $\mathbf{G}(\mathbf{q})$ is the gravity vector, which accounts for the torques induced at each joint due to the weight of the manipulator's links under gravity. $\boldsymbol{\tau}$ is the joint torque vector, representing the torques applied by the actuators.

The main concept of dynamics used in this thesis is inverse dynamics, which is used to compute the joint forces and torques required to generate a given motion. This is particularly useful for determining the torques necessary to execute a trajectory or maintain a static configuration. The inverse dynamics problem is solved using the recursive Newton-Euler (RNE) algorithm [28], which provides a computationally efficient approach for evaluating joint torques.

2.2 Direct Design and Performance Indicators

The use of computer software and algorithms to automate the creation, modification, analysis, or optimization of designs is often referred to as design automation. Automated design is commonly categorized into two approaches: direct design and inverse design (Fig. 2.3). Direct design follows a traditional forward process where engineers specify design parameters (e.g., dimensions, materials, actuator choices) based on heuristics, experience, or iterative simulations. The resulting system is then evaluated for performance, and adjustments in the design parameters are made as needed. Inverse design, on the other hand, takes a goal-oriented approach. Instead of specifying design parameters directly, desired performance criteria or constraints (e.g., payload capacity, workspace volume, maximum end-effector deflection) are defined first. Optimization algorithms then search for the best set of design parameters that achieve these objectives. If the resulting design does not satisfy expectations, the optimization problem can be adjusted.

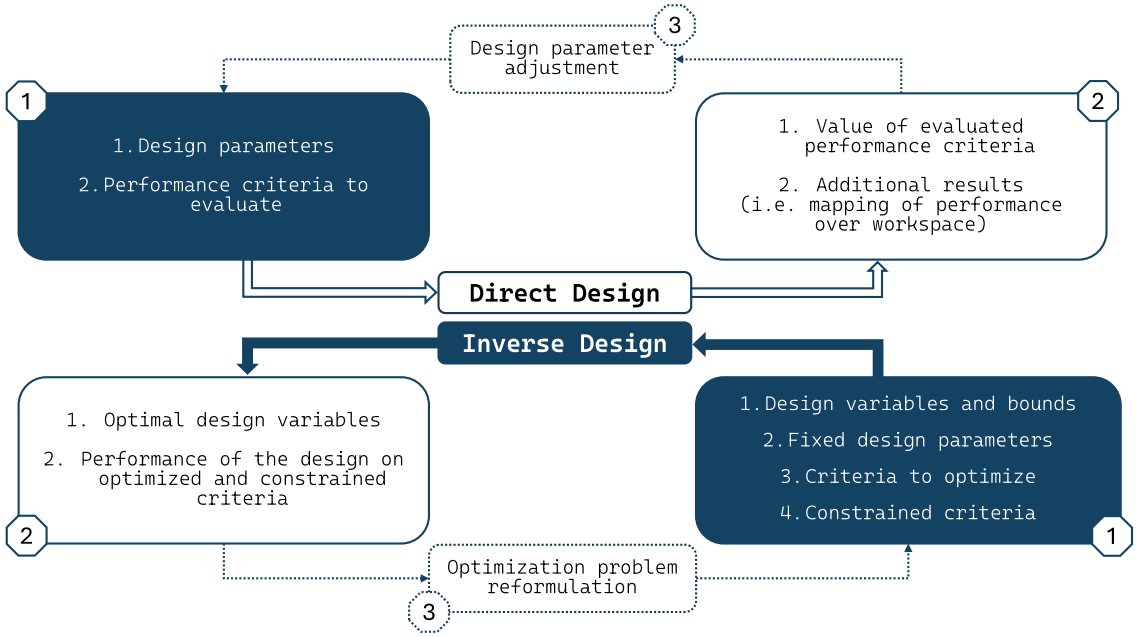


Figure 2.3 Illustration of the direct and inverse design processes with respective inputs and outputs.

The direct design of a robot can be distilled into a simple question : How good is a robot? [29]. While seemingly simple, the answer depends on the chosen performance indicators, which vary significantly based on the evaluation criteria. In robot data-sheets [30–33], industrial robotics companies tend to favor global, easy-to-interpret indicators such as payload capacity or reach, whereas researchers have developed a broad range of metrics tailored to specific tasks

and objectives. Although comprehensive indicators for the evaluation of mechatronic systems have been developed [34], this section focuses specifically on performance metrics tailored to the design and analysis of serial robot manipulators.

Several surveys have been conducted to compile existing performance indicators. The most recent review [29] highlights widely used metrics, while earlier surveys provide more comprehensive collections [35, 36]. For this master's thesis, only the most commonly used performance indicators for serial robot manipulator design in industry will be considered.

Recent surveys [29,36] agree on a common classification of performance metrics, distinguishing between local and global indicators, as well as kinematic and dynamic metrics. Local metrics depend on the robot's configuration, while global metrics evaluate performance across the entire workspace. Kinematic metrics assess motion-related properties such as reachability, manipulability, and dexterity, which are independent of forces and torques. In contrast, dynamic metrics incorporate the effects of mass, inertia, and applied forces, providing a more comprehensive evaluation of a robot's physical capabilities. Both types of metrics are crucial for assessing the efficiency, precision, and versatility of a manipulator, guiding both design decisions and performance comparisons. The following sections present an overview of the most commonly used indicators in both research and industrial applications.

2.2.1 Kinematic Metrics

Kinematic metrics constitute a large portion of the performance indicators studied in the literature [29, 35, 36], they describe the geometric and motion capabilities of a robot, independently of inertia and forces.

Workspace Metrics

Significant research has been devoted to analyzing the workspace of robotic manipulators. The reachable workspace of a robot is typically defined as the set of points that can be reached with at least one valid orientation [37]. While geometric and analytical approaches have been proposed to define the boundaries of a robot's reachable workspace [38], the most commonly used methods are based on sampling techniques [39, 40]. These methods sample the robot's joint space and discretize the Cartesian space into a grid of voxels. Forward kinematics is then applied to map sampled joint configurations to voxels. Finally, the volume of the reachable workspace is estimated as the sum of the volumes of all reachable voxels. Although sampling-based methods are efficient for the mapping and volume estimation of the reachable workspace, they suffer from a few limitations. Their precision depends on the granularity

of the discretization, and they are inherently stochastic, as the results are influenced by the random samples selected from the joint space [38].

The reachable workspace volume is highly dependent on the physical size of the robot, making it less relevant as a standalone measure of design efficiency. To address this, the Structural Length Index (SLI), a dimensionless indicator was developed to characterize the efficiency of the design with respect to the workspace volume [41]. It is defined as the ratio of the total link length of the manipulator to the cube root of its reachable workspace volume (Eq. 2.3). This index tends to be smaller for robots that have shorter link lengths relative to their reachable workspace volume, indicating a more efficient use of space in the robot's design.

$$SLI = \frac{\sum_{i=1}^{n} l_i}{\sqrt[3]{V}} \tag{2.3}$$

Where l_i is the length of the i-th link and V is the volume of the reachable workspace of the manipulator.

Although the previously discussed indicators are global, reachability is often a key factor in task-specific robot design, where the robot is intended to operate within a specific workspace or follow a predefined path. In these cases, the task is represented by a set of target positions or poses, and the goal is to quantify how well the robot can reach these positions.

Reachability refers to the existence of a robot configuration $\mathbf{q}$ that allows the end effector to attain the target pose. A common approach to determining whether a pose is attained is to check if the error between the target pose and the compared pose is within a specified tolerance. The error can directly serve as the reachability metric, with a smaller error indicating better reachability. Often, this metric is simplified to a binary indicator that outputs 1 if the error is within tolerance and 0 otherwise [42].

The configuration of the robot that corresponds to the minimum error is a result of inverse kinematics. The standard approach consists in running an IK algorithm for each target pose to obtain the configuration that minimizes the error [43–46]. Alternatively, some optimization studies treat the joint configurations required to reach each pose as design variables, simultaneously optimizing both the geometric variables and the joint configurations [11, 14, 47].

Earlier approaches used analytical methods for inverse kinematics, which resulted in configurations with imaginary components for unreachable poses [40]. In these cases, reachability was determined based on the magnitude of the imaginary component: a zero imaginary part indicated that the pose was reachable, while a nonzero magnitude represented the degree to which the robot was unable to reach the pose. Some studies further extend the definition of reachability by adopting a task-specific binary definition based on path planning,

where a pose is considered reachable if a collision-free path can be planned from a reference configuration to the pose [48–50].

The reachability indicator R for a task is commonly expressed as the ratio of reachable poses $N_{reachable}$ to the total number of poses N [51–53], where a value of 1 indicates that all poses are reachable (Eq. 2.4).

$$R = \frac{N_{reachable}}{N} \tag{2.4}$$

While reachable workspace volume and the reachability indicator are good indicators to describe the workspace, they lack information on the orientations with which the robot can reach each point within the workspace. To address this, additional indicators have been developed to describe the robot's dexterity, which refers to its ability to reach a given point with multiple orientations.

Different methods can be used to determine the orientations that can be achieved at a given point. The orientation space can be discretized evenly according to roll, pitch and yaw angles [54] or using uniformly spaced out points on an orientation sphere with a discretization around the third axis [55]. Inverse kinematics is then applied to each orientation to verify if it can be achieved. Leibrandt et al. [56] proposed a sample based method where the cartesian space is discretized into equal-sized 6D voxels and reachability of a each pose is evaluated by sampling the joint space. For all these different methods, the dexterity index is the ratio of achievable orientations over total tested orientations for a specified 3D point. This index allows the mapping of dexterity over the workspace (Fig. 2.4). It is a local indicator but can be used to assess the volume of the dexterous workspace, subregion of the reachable workspace composed of points where all orientations are achievable.

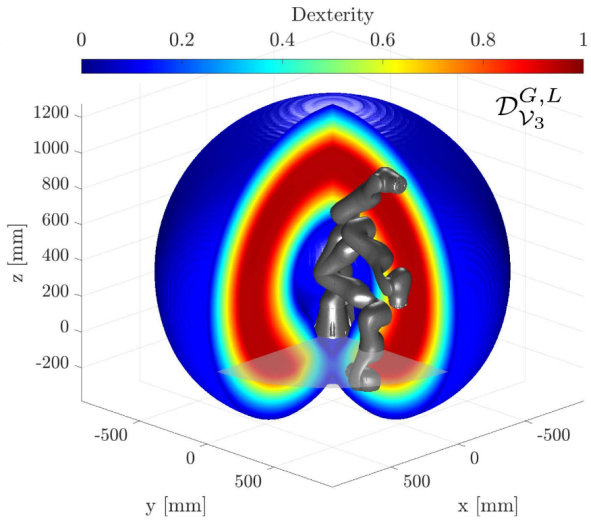


Figure 2.4 Mapping of the dexterity index over the workspace of a KUKA LBR iiwa 7 robot [31], with warmer colors representing a higher proportion of reachable orientations. [56]

Manipulability Metrics

Although the previous metrics offer valuable insights into a robot's workspace, they do not provide information regarding the singularities within that workspace. As discussed earlier, singularities occur when the Jacobian matrix of a robot loses rank. Manipulability metrics aim to assess a robot's ability, in a given configuration, to move and apply forces in various directions. Yoshikawa [57] introduced the manipulability index μ (Eq. 2.5), which is considered an effective measure of the robot's speed and force capabilities in all directions. However, this index has limitations, including its lack of boundedness and its dependence on units, joint types, and the number of DoF of the manipulator.

$$\mu = \sqrt{\det\left(\mathbf{J}\mathbf{J}^{\mathbf{T}}\right)} \tag{2.5}$$

Kim and Khosla [58] addressed the scale and order dependencies by modifying the order of the root and normalizing the index with a function of the square of the manipulator's length, denoted as f_m proposing a new index M_r (Eq. 2.6).

$$M_r = \frac{\sqrt[m]{\det\left(\mathbf{J}\mathbf{J}^{\mathbf{T}}\right)}}{f_m} \tag{2.6}$$

Building upon manipulability metrics, the conditioning number k is another important measure that provides insights into the robot's kinematic performance. Defined as the ratio

of the largest singular value σ_{max} to the smallest singular value σ_{min} of the Jacobian (Eq. 2.7), it indicates how errors in the joint space are amplified in the Cartesian space. This metric is an indicator of the kinematic isotropy of the Jacobian, providing a complementary perspective on the robot's ability to move efficiently along the major and minor axes of the manipulability ellipsoid.

$$k = \frac{\sigma_{max}}{\sigma_{min}} \tag{2.7}$$

This index is often preferred in the literature because it conveniently ranges from one to infinity and is a better measure of the degree of ill-conditioning of the manipulator [29]. Nevertheless, it still suffers from some of the limits of the manipulability index, scale dependency and non-homogeneity of the Jacobian.

Although these two indicators are local metrics, evaluating their average value over the joint-space $\mathcal{Q}$ provides a good indicator of a robot's manipulability over its workspace. The global manipulability index (GMI) (eq. 2.8) [36] and the Global Conditioning Index (GCI) (eq. 2.9) [59] were developed to provide global indicators.

$$GMI = \frac{\int_{\mathcal{Q}} \mu \, d\mathcal{Q}}{\int_{\mathcal{Q}} d\mathcal{Q}} \tag{2.8}$$

$$GCI = \frac{\int_{\mathcal{Q}} \frac{1}{k} \, d\mathcal{Q}}{\int_{\mathcal{Q}} d\mathcal{Q}} \tag{2.9}$$

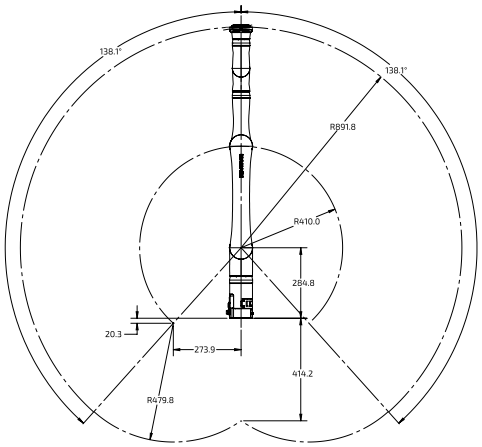
In the literature, the GCI is usually preferred due to its boundedness between 0 and 1 and the fact that it is independent from the number of DoF of the robot.

While kinematic performance metrics have been extensively studied in the literature, commercial data-sheets [30–33] typically emphasize three main indicators to characterize a robot's kinematic capabilities:

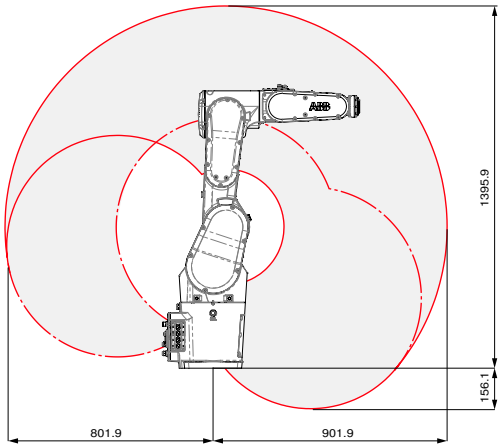
- Joint ranges [deg]: The lower and upper limits of each joint in the kinematic chain, defining the robot's motion capabilities.
- Workspace diagrams : Visual representations of the robot's reachable workspace. (Fig. 2.5)
- Reach [m] : Maximum distance the end-effector can extend in a specified direction.

Among these indicators, reach stands out as a scalar quantity, facilitating direct compari-

son between different robotic systems. In contrast, joint ranges and workspace diagrams, although highly informative, are less amenable to direct use in optimization processes.



(a) Workspace diagram of the Kinova Gen3 collaborative robot. [30]



(b) Workspace diagram of the ABB IRB1300 industrial robot. [33]

Figure 2.5 Workspace diagrams illustrating the reachability of a collaborative robot (Kinova GEN3) and an industrial robot (ABB IRB1300).

2.2.2 Dynamic Metrics

While kinematic metrics provide essential insights into the capabilities of a robot with respect to its workspace and singularities, understanding a robot's dynamic performance is equally crucial for evaluating its effectiveness in real-world applications. Dynamic metrics aim to assess the robot's performance considering its mass properties and sometimes consider the limits of its actuators.

The total mass of the robot is a key dynamic indicator, specifically important for collaborative or assistive robots where mass should be minimized. It is often compared with the payload capacity, maximum load the robot can hold at its end effector over the workspace, to measure a payload-to-weight ratio.

Other dynamic metrics are often specific to a path or a trajectory. The performance of a robot on a trajectory can be measured with the cumulative sum of torques produced at each actuator to complete the trajectory, the total energy consumption, or even the minimum cycle time to complete a task. These metrics are used in the manipulator optimization literature (see Section 2.3) to optimize the dynamic performance of robots.

Although less common than kinematic metrics, multiple dynamic indicators have been de-

veloped in the literature. Dynamic manipulability [60] measures the capacity of a robot manipulator to generate accelerations in all directions for a selected configuration. The dynamic conditioning index [61] defines the distance of the robot's general inertia matrix from dynamic isotropy, it is developed with the aim to facilitate control and simulation of a robot arm.

In commercial data-sheets, the most common dynamic metrics aim to be independent of a specific task:

- Total mass [kg]
- Payload capacity [kg]
- Maximum joint speed [rad/s]
- Maximum end effector cartesian speed [m/s]
- Average power consumption [W]

These metrics offer a general overview of a robot's dynamic characteristics but provide limited insight into performance for specific tasks.

2.2.3 Other Metrics

Several additional indicators are used to assess the performance of a robot manipulator. Standardized norms, such as ISO 9283 [62], are commonly used to evaluate the repeatability and precision of a robot over predefined trajectories. These results are critical for determining the suitability of the manipulator for specific tasks. For instance, surgical robots and precision assembly manipulators must demonstrate exceptional precision and repeatability to ensure consistent and reliable performance.

Another important metric is the robot's range of working temperatures, which indicates its suitability for various operational environments. This metric is particularly relevant for robots deployed in extreme conditions, such as space exploration or industrial settings with temperature fluctuations.

Stiffness-related metrics also play a significant role in robot performance analysis. While robot stiffness depends on both link stiffness and joint stiffness, the latter is typically the dominant contributor to compliance [63]. As a result, link stiffness is often neglected in stiffness evaluations, leading to the derivation of a diagonal stiffness matrix $\mathbf{K}_\theta$ that captures

the torsional stiffness of each joint. The joint angle deflection $\Delta\boldsymbol{\theta}$ due to joint torques $\boldsymbol{\tau}$ is expressed as:

$$\mathbf{K}_\theta \Delta\boldsymbol{\theta} = \boldsymbol{\tau} \tag{2.10}$$

Using this joint-space stiffness matrix, various indicators can be derived to assess maximum deflection [64] or Cartesian stiffness [65]. Additionally, a linearized dynamic model can be obtained at a given joint configuration using the stiffness matrix and the inertia matrix of the robot $\mathbf{M}$, as shown in the following equation (Eq. 2.11):

$$\mathbf{M}\Delta\ddot{\theta} + \mathbf{K}_\theta \Delta\theta = \boldsymbol{\tau} \tag{2.11}$$

This represents a typical free vibration equation, where $\Delta\theta$ denotes the displacement from equilibrium. The natural frequencies of the robot can then be determined by solving the generalized eigenvalue problem:

$$(\mathbf{K}_\theta - \omega^{\mathbf{2}}\mathbf{M})\boldsymbol{\Phi} = \mathbf{0} \tag{2.12}$$

Here, ω represents the natural frequency, and $\boldsymbol{\Phi}$ denotes the corresponding mode shape. The minimum natural frequency is a critical performance indicator, as it defines the range of frequencies that may arise during operation. Ensuring that the manipulator's natural frequencies do not coincide with excitation frequencies in the working environment is essential to avoid resonance, which could lead to undesirable vibrations and reduce the robot's performance.

Table 2.1 provides an overview of the presented metrics, specifying whether they are local or global or task-dependent and the physics considered.

Table 2.1 Summary of commonly used manipulator performance metrics in scientific literature and commercial data-sheets.

Metric	**Unit**	**Local/ Global**	**Task**	**Physics**
Workspace Volume	m^3	global		kinematics
SLI	-	global		kinematics
Reachability Indicator	% or m	global	x	kinematics
Dexterous Workspace Volume	m^3	global		kinematics
Manipulability Index	-	local		kinematics
Conditioning Number	-	local		kinematics
GMI	-	global		kinematics
GCI	-	global		kinematics
Reach	m	global		kinematics
Mass	kg	global		dynamics
Sum of Torques	Nm	global	x	dynamics
Energy Consumption	J	global	x	dynamics
Min. Cycle Time	s	global	x	dynamics
Payload Capacity	kg	global		dynamics
Maximum End-Effector Cartesian Speed	m/s	global	x	dynamics
Average Power Consumption	W	global	x	dynamics
Natural Frequency	Hz	local		dynamics
Repeatability	mm	global	x	control
Working Temperature Range	degrees	global		thermodynamics
Deflection	mm	local		structural mechanics
Stiffness	Nm/rad or N/m	local		structural mechanics

2.3 Inverse Design and Optimization

Having reviewed the broad spectrum of metrics used to characterize robotic systems, this section now focuses on the inverse design of robot manipulators, a subject that lies at the core of this thesis. In this section, the state-of-the-art optimization methodologies that have been proposed in the literature are critically examined. These studies address global and task-specific criteria, ranging from kinematic and dynamic performance to optimal drive-train selection.

The literature on robot optimization is vast and diverse. Over the years, researchers have studied the optimization of robotic manipulators for numerous tasks and applications, constructing a variety of optimization problems. The following review will be structured by studying the formulation problems and optimization structures used in the literature.

The general structure of an optimization problem consists of four key components. The first is the objective function $f(\mathbf{x})$, which is to be either minimized or maximized. Optimization problems may also include constraints, in which case the objective function is optimized while ensuring the feasibility of all equality constraints, $\mathbf{h}(\mathbf{x})$, and inequality constraints, $\mathbf{g}(\mathbf{x})$. A design parameter (DP) of the robot that can be varied and optimized during the process is referred to as a design variable (DV). Design variables are grouped within a vector $\mathbf{x}$ that belongs to a domain $\mathcal{X}$, representing the allowable bounds for each variable. Finally, optimization problems can be approached using various optimization structures and algorithms, depending on the problem formulation and computational requirements. The general form of an optimization problem is presented in Equation 2.13.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, m, \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, p, \\ & \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{2.13}$$

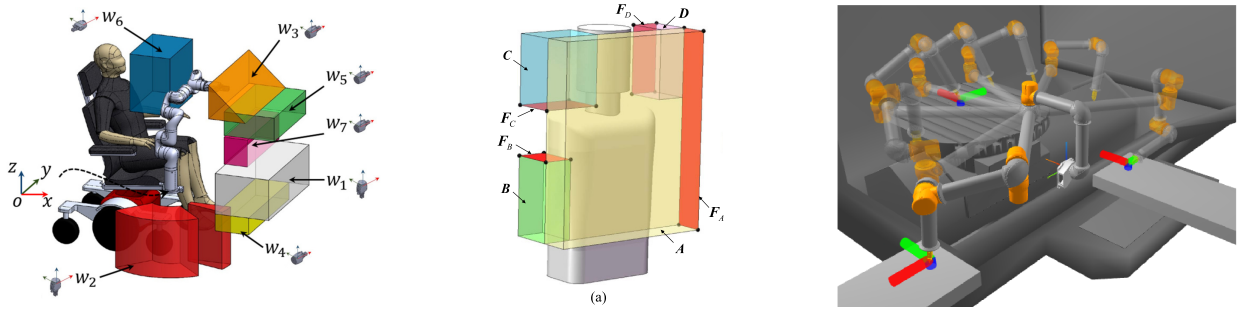
The following sections review the literature by first examining kinematic optimization, followed by dynamic optimization, and then approaches that integrate both kinematic and dynamic considerations. Next, studies incorporating structural optimization are discussed, before transitioning to an analysis of optimization structures and the algorithms used to solve these problems.

2.3.1 Kinematic Optimization

In the literature, kinematic optimization was one of the first extensively studied type of optimization for robotics [40, 59, 66], focusing on refining a robot's geometry to enhance reachability, dexterity, or manipulability, without accounting for inertia and dynamic effects.

In the design of robotic manipulators for specific applications, reachability is a fundamental performance criterion. As a result, optimizing reachability was among the earliest challenges addressed in robotic design optimization [40], with initial studies focusing on determining the optimal link dimensions to access a predefined set of target points. Early research primarily considered two-dimensional task spaces, but later studies expanded this approach to encompass three-dimensional task poses [43]. This class of optimization problems is applicable to a wide range of domains, including assistive robots [42], humanoid robots [50] or industrial robots [11, 67], maximizing the reachability of application specific task spaces (Fig. 2.6a and Fig. 2.6b) or poses (Fig. 2.6c).

The design variables used to optimize reachability vary, ranging from individual link lengths to the complete set of DH parameters and joint types. Kawaharazuka et al. [46] consider all of these factors, investigating the trade-offs between design complexity and the ability to reach a given set of poses. The optimal solutions converge towards both existing and novel joint configurations, demonstrating the effectiveness of inverse design methods for exploring innovative concepts.



(a) Objective task space for an assistive robot. [42]

(b) Objective task space for a humanoid robot. [50]

(c) Pick-and-place task poses for an industrial robot. [67]

Figure 2.6 Representations of volumes (a-b) and poses (c) used as objective task spaces for different assistive, humanoid and industrial robot applications.

A key limitation of reachability, as previously defined, is that it only guarantees the existence of a robot configuration capable of reaching the task without accounting for whether a feasible path exists to achieve it. To address this, several studies redefine reachability by considering

the existence of a collision-free path between a reference configuration and the configuration required to perform the task. Optimization approaches tackling this challenge typically fall into two categories: those that incorporate path planning as a subproblem at each step of the optimization process [67] and those that directly treat sets of joint angles as design variables [48, 68]. Zhu et al. [50] extend this perspective for humanoid robots by identifying optimal link lengths for a human-like reachable workspace, using a quadratic programming controller to generate collision-free paths toward each task point.

Although such optimization techniques yield robotic manipulators tailored for specific tasks, some robotic systems are designed without a predefined application. In these cases, the optimization objective may instead focus on maximizing the volume of the reachable workspace by adjusting link lengths [41, 69] or by optimizing the complete set of DH parameters [70, 71] within given ranges. With the same aim, Galan-Uribe et al. [72] optimize the SLI to ensure an efficient workspace-volume-to-robot-size ratio. Although these optimization problems result in robots with a larger workspace, information on the orientation of the end effector across the workspace is omitted. Studies have proposed to maximize the dexterous workspace volume instead in order to obtain a robot able to reach a large volume with any orientation [56, 73].

While optimizing for reachability ensures that the manipulator can access designated task points, it does not account for the robot's ability to perform agile movements at those points. Specifically, a critical question arises: can the robot move efficiently in all required directions from a given pose? Additionally, how effectively can it transform joint torques and velocities into forces and velocities at the end effector? To address these concerns, researchers have explored the optimization of manipulability throughout the entire workspace, utilizing metrics such as the GCI [41,66,72], as well as the optimization of manipulability for specific tasks [74] by varying link lengths or modular components [75].

Given the complementary nature of these optimization criteria, several studies have sought to integrate them into a more general kinematic optimization problem.

One common approach is to optimize manipulability while enforcing reachability constraints for a given task [76]. Alternatively, both criteria can be incorporated into a single aggregated objective function [77]. Chocron [78] further extends this methodology by considering additional design objectives, such as minimizing the manipulator's size and maximizing the distance to an obstacle, within the aggregated function. Franceschi et al. [79] consider the problem from another angle, instead of optimizing the robot's structure, the placement of the task within the workspace is optimized with respect to collision free path existence and manipulability.

In a broader context, global workspace optimization has been performed by simultaneously

maximizing workspace volume and global manipulability indices. Several studies have used multi-objective optimization techniques to optimize workspace-related metrics, incorporating indicators such as the SLI and the GCI to balance reachability and manipulability [80–82]. Similarly, Cursi et al. [83] propose to optimize the global kinematic design by maximizing the volume of the manipulable workspace, a subregion of the reachable workspace defined by a manipulability index threshold.

2.3.2 Dynamic Optimization

Rather than prioritizing kinematic performance, several studies have focused on optimizing the dynamic behavior of manipulators. These studies can be broadly classified into two main approaches: those that seek to enhance the robot's performance for a specific task and those that aim to improve task-independent robot characteristics while ensuring the feasibility of a given task.

Quantifying the performance of a manipulator over a task can take different forms. Minimizing the actuating torques necessary to achieve a trajectory is a common approach. In this aim, some studies vary the mass repartition of the links [84–86]. Toussaint et al. [87] consider the impact of link deformation modifying the length and orientation of links to minimize the torques over a trajectory. Similarly, Fadini et al. [88] attempted to minimize the energy consumed to perform a selection of tasks. Another approach is to attempt to minimize the time it takes for the manipulator to perform a task. This criteria is often combined with minimizing torques [89], minimizing energy consumption in actuators [90] or both [91] in order to find a good compromise between speed and energy efficiency.

The parallel approach consists in considering task feasibility as a constraint and optimizing other manipulator performance criteria. This type of optimization often includes the joint actuators as design variables, as the feasibility of a reachable task is usually limited by the performance of the actuators. A possible process is to minimize the mass of the drive train by selecting the lightest motors, gearboxes [92] or assembled drive trains [93] capable of producing the necessary torques and speeds to complete the task. When selecting drive train components from a catalog, the cost of each item is usually specified and the total cost can therefore be minimized [94, 95]. Some authors have also proposed to model the relationship between mass, available torque, inertia and other dynamic properties of actuators to find the optimal actuator to perform a task [96, 97]. This approach can lead to superior results but is often avoided because of the necessity to design custom actuators.

The dynamic optimization problem can also be formulated to simultaneously optimize the task performance and the manipulator, minimizing cycle time and cost [98], cycle time and

mass [99], energy consumption and mass [100], or minimizing cycle time and maximizing natural frequency [101].

2.3.3 Kinematic and Dynamic Optimization

The performance of a robotic manipulator cannot be fully characterized by considering only kinematic or only dynamic factors. For instance, a robot optimized for reachability and manipulability over a given task may still fail to execute it if its actuators cannot generate the required torques due to the absence of dynamic considerations. Conversely, a manipulator optimized solely for minimal mass, leading to shorter link lengths, may achieve efficient dynamic performance but suffer from reduced manipulability [102], making it highly sensitive to perturbations.

To address these limitations, optimization problems are often structured to incorporate both kinematic and dynamic considerations. For example, Sanjuan de Caro et al. [44] simultaneously optimize the reachability of an assistive robot's workspace and the static torques required to maintain equilibrium in every configuration.

Another optimization problem is minimizing the mass of a manipulator while ensuring trajectory feasibility. It can be approached as a purely dynamic optimization problem if the actuators are considered as the only design variables. However, further mass reduction can be achieved by shortening link lengths but it requires an additional constraint: the manipulator must still be capable of reaching all required task points [103]. Zhu et al. [49] extend this idea by incorporating both mass and manipulability optimization to determine optimal actuator selection and link lengths for a humanoid arm designed for driving. Their formulation results in a lightweight arm capable of executing trajectories with high manipulability and low joint torques. Likewise, Hoffman et al. [14] optimize actuator selection and link lengths to maximize reachability and manipulability while minimizing the static torques required to access all points within a specified workspace.

As previously discussed, some robots are designed to perform optimally across their entire workspace, with the GCI and SLI serving as key indicators of kinematic performance. Hwang et al. [104] and Li et al. [105] integrate these metrics with an evaluation of the dynamic conditioning number along a predefined path, leading to designs that achieve both high kinematic and dynamic performance. Alternatively, Zhou et al. [106] adopt a different strategy by minimizing the manipulator's mass, selecting optimal actuators and link lengths to ensure trajectory feasibility while imposing the GCI as a constraint. A challenge with this approach lies in determining an appropriate constraint value, as the GCI can be difficult to interpret.

2.3.4 Structural Optimization

While kinematic and dynamic optimization enhance motion performance, actuator efficiency, and task feasibility, these factors alone do not guarantee a well-performing robot. Structural properties such as deflection, stiffness, and strength are equally important in ensuring reliability, precision, and durability. A robot optimized for kinematic and dynamic performance may struggle with control if its links and actuators lack sufficient stiffness to prevent excessive end-effector deflection. Similarly, minimizing mass by selecting lightweight actuators and shorter links can be beneficial, but overly short links may compromise the robot's ability to reach its intended tasks. In such cases, adjusting the structural properties of the links could provide a solution, but this must be done while considering material strength and deflection constraints. For these reasons, many studies have integrated structural optimization into the design process, aiming to improve mechanical robustness while preserving optimal kinematic performance, dynamic performance or both.

In manipulator optimization literature, structural considerations are typically included as constraints, such as limits on end-effector deflection or maximum allowable stress [53, 107–109], or as objectives, such as minimizing deflection [110–113] or maximizing stiffness [64, 65]. As previously discussed, structural performance is primarily influenced by link stiffness and joint stiffness. Link stiffness depends on material properties and geometry, with design variables including cross-sectional parameters for complex geometries [108, 110, 114] or link length, shell thickness, and diameter for cylindrical approximations [53, 107, 109, 112, 113]. Joint stiffness is often the dominating factor for the overall stiffness of a manipulator, particularly when using harmonic drives due to their inherent flexibility. Consequently, some studies have limited the stiffness modeling to the torsional rigidity of each actuator [64, 65]. Du et al. [115] integrate both link and joint stiffness in an optimization problem that maximizes the mass-to-payload ratio and natural frequency.

Although structural optimization can be considered independently [111], it is often integrated with kinematic and dynamic considerations to achieve a well-balanced design. Several studies use global optimization to jointly improve manipulability and stiffness [65], maximize natural frequency while minimizing deflection [112], or minimize mass while imposing constraints on manipulability, stress, and deflection [109]. In task-specific designs, optimization frameworks have been developed to minimize actuator torques while enforcing deflection constraints [108, 114] or maximize stiffness while satisfying manipulability requirements [64].

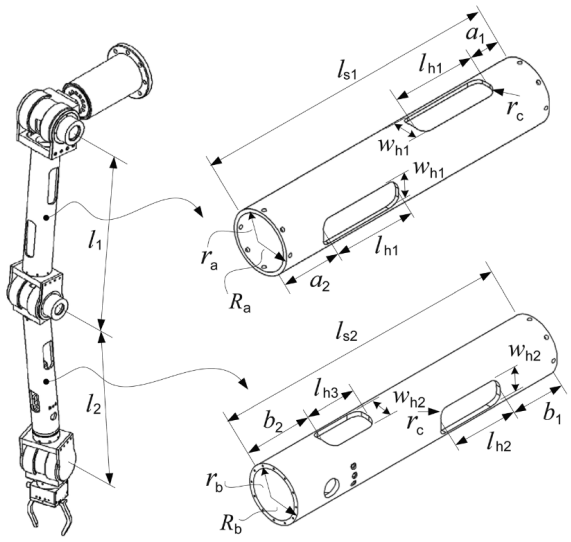


Figure 2.7 Example of structural design variables for manipulator mass minimization with deflection constraints. [109]

2.3.5 Optimization Structure

While kinematic, dynamic, and structural considerations define the objectives of robot design optimization, the way these objectives are formulated and solved significantly impacts the final outcome. Different optimization structures offer distinct approaches to balancing competing design criteria. The following section explores the most common methodologies, highlighting their advantages, limitations, and applications in the optimization of robotic manipulators.

One of the key considerations when constructing the manipulator optimization problem is whether to consider the various performance criteria as objective functions or constraints. Indeed, performance criteria often compete with each other and a choice has to be made for which ones to pose as constraints with appropriate thresholds and which one to optimize to reach a maximum or minimum. Constructing a simple unconstrained problem has been studied [71, 73], but the most complete manipulator optimization problems include multiple constraints and sometimes multiples objectives.

A common approach is to identify a single criteria to use as an objective function and constrain the problem with the other criteria. For example, Zhou et al. [109] choose to minimize the mass as the objective and consider constraints on the minimum GCI, maximum stress and maximum deflection as well as drive train torque and speed limits.

Multi-Objective Optimization

In certain cases, design optimization requires selecting multiple objective functions simultaneously. To address this challenge, two main approaches are commonly used: aggregated functions and multi-objective optimization. Aggregated functions simplify the problem by combining multiple objectives into a single function, typically using weighted sums (Eq. 2.14) or other scalarization methods. This approach allows traditional optimization techniques to be applied but requires careful selection of weights to properly balance competing objectives. Hoffman et al. [14] use the weighted sum of reachability error, trajectory torques and manipulability to derive an unconstrained optimization problem. This method allows the use of optimization algorithms that deal poorly or cannot deal at all with constraints. Xu et al. [65] use a different form of aggregated function, a quotient of manipulability and stiffness indexes to create a novel global performance index to optimize an anthropomorphic manipulator. This approach can be used to optimize a great number of functions simultaneously, Chocron and Bidaud [78] use a a weighted sum within an exponential function to optimize reachability, maximum reach, obstacle proximity and manipulability and minimize the number of modular components for a modular robot. However, a key challenge with this approach is that different functions may have vastly different scales and behaviors, making weight selection highly sensitive and difficult to anticipate.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_{aggregated}(\mathbf{x}) = \omega_1 f_1(\mathbf{x}) + \omega_2 f_2(\mathbf{x}) + \omega_3 f_3(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{2.14}$$

Where $f_{aggregated}$ is the aggregated function combining functions f_i with respective weights ω_i.

Another aggregation technique is the minimax method, which focuses on minimizing the worst-performing objective within a set of functions $\mathbf{f}$ (Eq. 2.15). Castejón et al. [110] use this technique to optimize scaled functions for maximum reach, workspace volume, mass, stiffness and safety to design a service robot.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_{minimax}(\mathbf{x}) = \max(\mathbf{f(x)}) \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X} \\ \text{where} \quad & \mathbf{f} = \begin{bmatrix} f_1(\mathbf{x}) & f_2(\mathbf{x}) & f_3(\mathbf{x}) \end{bmatrix}^{\mathbf{T}} \end{aligned} \tag{2.15}$$

Alternatively, Multi-Objective Optimization (MOO) techniques aim to find a set of non-dominated designs known as the Pareto front, where improving one objective cannot be

achieved without degrading another (Eq. 2.16). The resulting set of optimal solutions allows engineers to visualize the trade-offs between different performance criteria and select the design that provides the better compromise. MOO has been used to explore trade-offs between reachability and static torques [45] , mass and deflections [113], and mass and workspace volume [53]. The results of such studies not only improve on the original design but also provide a selection of candidate designs with different levels of trade-off between the objective functions.

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f(x)} \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X} \\ \text{where} \quad & \mathbf{f} = \begin{bmatrix} f_1(\mathbf{x}) & f_2(\mathbf{x}) & f_3(\mathbf{x}) \end{bmatrix}^{\mathbf{T}} \end{aligned} \tag{2.16}$$

Handling Discrete Variables

Another complexity when constructing an optimization problem can arise when dealing with discrete variables. The most common discrete variable used in robot manipulator optimization is actuator selection. Although some studies choose to develop continuous actuator models [88,90,96], the resulting designs are often unrealistic as they require designing custom motors and gearboxes. Most robotic companies design their actuators by selecting components from specialized motor and gearbox catalogs. Therefore, optimization problems that deal with discrete variables are often preferred. The most common approach is to conserve the classic structure of an optimization problem (Eq. 2.13), including the discrete design variables and use an algorithm that can simultaneously optimize discrete and continuous variables. This category of optimization is referred to as mixed-integer programming. Gulec and Ertugrul [113] combine MOO and mixed-integer programming to optimize the link thicknesses and motor and gearbox selection, identifying the trade-offs between mass and stiffness. From the resulting Pareto-front, the authors identify a set of shell thicknesses, motors and gearboxes for a lightweight design (3.4 kg) while limiting the end effector deflection to a reasonable value (1.1 mm).

Alternatively, discrete and continuous variables can be dealt with separately. Either by sequentially optimizing variables like link lengths then optimizing actuator selection [49] or by using a bi-level optimization structure. Bi-level optimization methods provide a structured approach for handling mixed-integer programming problems by separating discrete and continuous variables into distinct hierarchical levels. In such schemes, the upper-level problem typically involves discrete decisions, such as actuator selection or joint configurations, while the lower-level problem optimizes continuous variables, like link lengths or structural

parameters. This hierarchical structure allows for efficient decomposition of complex optimization problems, facilitating the handling of both types of variables. Hoffman et al. [14] use a bi-level structure to optimize actuator selection in the outer loop and link lengths and base placement in the inner loop, profiting from the optimization scheme to use an efficient algorithm for each type of variable.

Computational Efficiency

Due to the complexity and size of the design space, optimization algorithms often require the evaluation of functions across numerous combinations of design variable values. While this may not be problematic for inexpensive functions, certain performance criteria used to assess robot designs can be computationally expensive. To address this challenge, some studies use surrogate models, which replicate the behavior of the original functions but are more computationally efficient. For instance, Tian et al. [116] utilize a Kriging surrogate model to approximate the workspace volume and dexterous workspace volume functions. Similarly, Lim et al. [81] approximate the GCI and SLI with Kriging, achieving an error of less than 1%. In optimization problems that rely on external software, such as computer-aided design (CAD) or finite element analysis (FEA) tools to evaluate structural criteria, surrogate models can significantly reduce computation time [98].

For certain functions, computational efficiency can be further improved by implementing variable-accuracy models. Functions such as workspace volume evaluation, or any local indicator integrated over the workspace, generally exhibit increasing accuracy with the number of samples. However, achieving high precision can be computationally expensive. To mitigate this, Baykal et al. [48] propose varying the accuracy of the surrogate models during the optimization process, starting with lower accuracy functions for efficient exploration and progressively increasing the accuracy as the optimization converges.

2.3.6 Optimization Algorithms

Given the complexities of serial manipulator conceptual design and the challenges associated with evaluating performance criteria, various optimization algorithms have been used in the literature to navigate the design space effectively. These algorithms are crucial for identifying optimal configurations while considering a range of design constraints and objectives. This subsection provides an overview of commonly used algorithms in the literature.

Some studies have considered the exhaustive search of the design space, by discretizing continuous variables and storing the best result [93, 97]. Although some filtration techniques

allows to efficiently discard unfeasible designs [117], these techniques are usually not preferred as they require large computation power and are inefficient. Consequently, studies related to inverse conceptual design of manipulators use optimization algorithms to increase time-efficiency and reduce the necessary computation power. For the purpose of this review, optimization algorithms will be categorized into two main groups: gradient-based algorithms and gradient-free algorithms. Due to the wide variety of gradient-free algorithms, they will be further subdivided into evolutionary algorithms, swarm intelligence algorithms, direct search algorithms, and multi-objective optimization algorithms.

Gradient-Based Algorithms

Gradient-based algorithms leverage gradient information to efficiently explore the design space. They are well-suited for high-dimensional problems and particularly effective for optimizing smooth functions. Their fast convergence makes them ideal for convex and unimodal optimization; however, they are prone to getting trapped in local optima when applied to highly multi-modal functions and struggle with discontinuous objective functions. Despite these limitations, a few studies have used gradient-based approaches. Xu et al. [65] used the LM algorithm to optimize the link lengths of an anthropomorphic manipulator for improved manipulability and stiffness. Similarly, Kivelä et al. [11] applied the same algorithm to a high-dimensional optimization problem, incorporating a multi-start strategy to mitigate local optima. Their results revealed significant variations in the optimal variables, highlighting the challenges gradient-based methods face in multi-modal landscapes. Other studies have explored Sequential Quadratic Programming (SQP) for robotic design optimization [80,110]. Bergamaschi et al. [70] compared SQP with evolutionary algorithms for workspace volume maximization, finding that while SQP converges more rapidly, evolutionary methods are better at escaping local minima, often yielding superior solutions.

Evolutionary Algorithms

Evolutionary algorithms (EAs) are inspired by biological evolution, using mechanisms such as selection, mutation, and crossover to iteratively refine candidate solutions. These algorithms are particularly effective for optimizing complex, multi-modal, and discontinuous functions where gradient-based methods struggle. Genetic Algorithms (GA) were among the first methods applied to robotic design optimization [47, 114, 118] and continue to be used in modern approaches [53,67,101]. Some studies have explored Differential Evolution (DE) as an alternative to a GA, leveraging adaptive mutation strategies to achieve improved optimization performance in certain applications [70, 119]. More recently, Covariance Matrix Adaptation

- Evolution Strategy (CMA-ES) has gained attention for its ability to autonomously adjust search parameters, making it particularly well-suited for high-dimensional design spaces [49, 50, 88, 90].

Swarm Intelligence Algorithms

Swarm intelligence algorithms are inspired by the collective behavior of biological systems, such as flocks of birds or colonies of ants. These algorithms rely on a population of interacting agents that collaboratively explore the search space, making them particularly effective for highly multi-modal optimization problems. Among them, Particle Swarm Optimization (PSO) is one of the most widely applied methods, modeling the optimization process as a dynamic interaction between particles adjusting their positions based on both personal and global best solutions. However, multiple studies have systematically shown that PSO tends to under-perform compared to other algorithms in robotic design optimization [70, 72, 83]. Other swarm-based approaches have also been explored. Ant Colony Optimization (ACO), which simulates pheromone-based path-finding, has been successfully used in a bi-level optimization framework to efficiently select optimal actuators [14]. Additionally, more recent nature-inspired algorithms such as the Grey Wolf Optimizer (GWO) and Harris Hawks Optimization (HHO) have been compared to GA and PSO, showing promising results for kinematic optimization [72].

Direct Search Algorithms

Direct search methods are derivative-free optimization techniques that explore the search space using geometric transformations or structured patterns rather than relying on gradients. These methods are particularly useful for constrained problems and scenarios where function evaluations are expensive or noisy. Among them, the Complex method has been widely used, particularly in its extension to mixed-integer problems developed by Pettersson et al. [120]. This approach has been extensively applied at Linköping University for optimal drivetrain design, where it was used to select suitable gearboxes and motors for both industrial and modular robots [95, 96, 121]. More recent studies that incorporate actuator selection alongside other design variables have also adopted the Complex method, reporting excellent optimization performance [109, 122].

Alternative Approaches

Although the previously discussed algorithms are the most widely used in the literature, some studies have explored alternative approaches such as Simulated Annealing [48,76] or Bayesian Optimization (BO), which has emerged as a particularly promising method. It operates by constructing a probabilistic model of the objective function and strategically selecting new evaluations to minimize function queries, thereby significantly reducing computational cost. Cursi et al. [83] report that BO achieves performance comparable to that of a Genetic Algorithm (GA) while reducing computation time by a factor of six.

Multi-Objective Optimization Algorithms

As presented previously, many robotic design problems involve optimizing multiple conflicting objectives. Multi-objective optimization algorithms aim to generate a set of trade-off solutions known as the Pareto front. The most widely used method is Non-Dominated Sorted Genetic Algorithm II (NSGA-II), which extends the evolutionary principles to multi-objective spaces [99,102,113,115]. Saravan et al. [108] compare this method to Multi-Objective Genetic Algorithm and Multi-Objective Differential Evolution and reports that NSGA-II outperforms by far the others on a complete kinematic, dynamic and structural optimization problem.

2.3.7 Design Exploration and Optimization Tools

The previous sections have outlined the extensive body of literature on the design optimization of robot manipulators. While the studies presented tackle a wide variety of optimization problems, the methodologies used often lack the capacity for generalization across different optimization contexts. A few studies have made attempts to develop more general frameworks for robot design optimization, though these efforts remain limited.

One notable example is the GlobDesOpt framework developed by Cursi et al. [83], which focuses purely on kinematic optimization. This Matlab-based framework enables the optimization of DH parameters and joint types to maximize global manipulability for both single-arm and dual-arm robots. It incorporates implementations of Genetic Algorithms, Particle Swarm Optimization and Bayesian Optimization, offering flexibility in addressing different optimization challenges. However, it remains confined to kinematic optimization and does not extend to other critical aspects like dynamics or modular design.

Another relevant contribution is the work of Leger [123], which introduces Darwin2k, a software for the automated design of robot configurations. This research presents an evolutionary approach where robot designs are represented using parameterized module configuration graphs, allowing for both structural and kinematic design variables. The system incorporates a simulation environment with capabilities for dynamic analysis, collision-free path planning, and the evaluation of various performance metrics. Several case studies, including a free-flying space robot and a manipulator for a space shuttle, demonstrate the system's ability to synthesize effective and novel robot designs. The framework is primarily limited to task-based performance metrics and relies on a GA for optimization.

Finally, Külz et al. [124] developed Timor Python, a toolbox tailored for the assembly, design exploration and optimization of modular robots. It allows users to assemble robots from standardized modules and generates both kinematic and dynamic models. The optimization interface supports multiple objective functions, such as cycle time, power consumption, and robot mass, and uses a GA for optimization. However, its focus on modular robots prevents it from supporting continuous design variables. Additionally, the toolbox is constrained to specific task-based optimization problems.

2.4 Literature Review Discussion

In conclusion, while the literature on the design optimization of robot manipulators is extensive, it remains largely fragmented and focused on specific, often narrowly defined problems. Existing approaches lack the generality and flexibility needed to support the early-stage design of manipulators across diverse tasks and environments. This section highlights the main limitations in the literature, which arise from the narrow scope of applications considered, incomplete or oversimplified modeling assumptions, restrictive problem formulations, and algorithmic limitations.

A primary limitation is the incomplete modeling of the manipulator system. As discussed in Sections 2.3.1 and 2.3.2, many studies model only subsets of the design space, typically focusing on either kinematic parameters or actuator selection, without considering a comprehensive representation of the manipulator.

A second limitation is the task-specific nature of most methodologies. Many design strategies are tailored to a single application, which limits their generalizability. Conversely, methods that optimize global criteria often lack the ability to incorporate task-specific performance metrics. The developed formulation of the optimization problem itself is often restrictive, either lacking the consideration of essential requirements or limiting the analysis to a narrow scope.

Additionally, limitations arise from the algorithms used to solve these problems. Most studies only consider optimizing the problem with a single algorithm, and may miss out on the opportunity to find better solutions more efficiently with other algorithms. Because of their widely different characteristics, optimization algorithms perform differently for different problems. Some may deal with constraints better than others, some may naturally consider integer variables, and some may necessitate fewer function evaluations. A broader exploration of algorithmic strategies, tailored to the nature of the problem at hand, is often missing from the literature.

More broadly, the surveyed methods generally address isolated design problems rather than contributing to the development of a unified and versatile design optimization framework. The few tools that do attempt a more general approach are still subject to the aforementioned limitations.

2.5 Project Rationale

In conclusion, despite the significant advancements in the field of manipulator optimization, the available optimization tools for conceptual design remain few and limited in scope.

Thus, the research objective of this thesis is to develop a versatile optimization and exploration framework for the conceptual design support of serial robot manipulators. This objective is structured into three sub-objectives.

SO-1 - Select the most relevant performance evaluation metrics and model their relationship to design parameters

SO-2 - Develop a general framework for manipulator design optimization for various optimization problems.

SO-3 - Identify the most suitable algorithm for various optimization problems.

To achieve these objectives, Chapter 3 presents the methodology used to construct a general-purpose framework. This includes the development of a comprehensive manipulator model, the implementation of functions for evaluating performance metrics, and the formulation of a flexible inverse design optimization structure. Chapter 4 then applies the developed framework to construct a series of representative design problems and identify the most efficient algorithms. Figure 2.8 provides an overview of the developed framework, indicating which sections of this master's thesis correspond to each of its components.

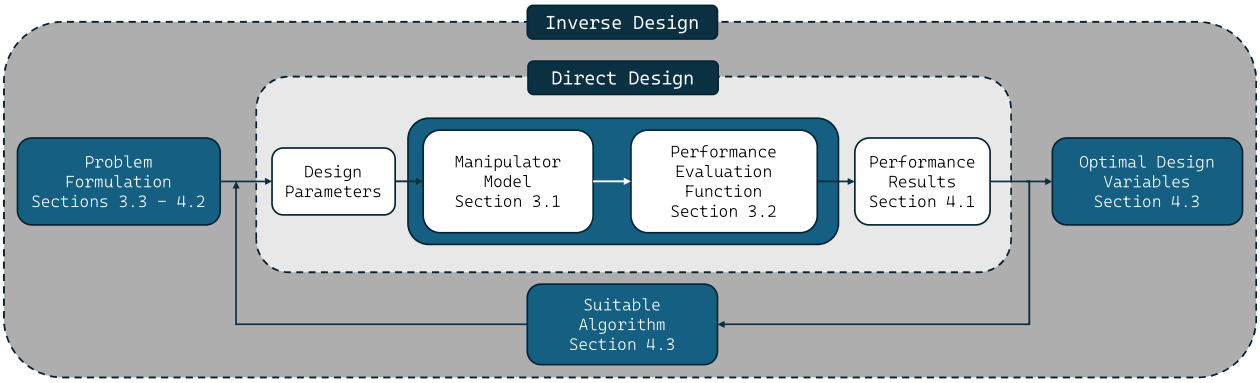


Figure 2.8 Overview of the direct and inverse design frameworks, highlighting the key components, their interrelations, and references to the corresponding sections of the thesis.

CHAPTER 3 METHODOLOGY

The conceptual design phase of robotic manipulators involves the exploration of diverse design concepts and the efficient evaluation of their performance with respect to predefined requirements. Traditionally, this process relies on the collaborative efforts of engineers from multiple disciplines to assess each concept against a range of performance criteria. However, for each new concept, the associated performance calculations and simulations often need to be redeveloped, making the process time-consuming and inefficient.

This chapter proposes a generalized framework that supports both direct and inverse design methodologies for evaluating and optimizing the performance of manipulator concepts. The first section introduces the parameterized model of a serial manipulator composed of revolute joints, enabling consistent representation across different design variations. The second section details the selection and implementation of performance evaluation functions, supporting the direct design approach by enabling rapid assessment of design concepts. Finally, the third section presents the formulation of an optimization problem that integrates the manipulator model and performance criteria, thereby enabling an inverse design approach to systematically search for optimal design configurations.

3.1 Modeling and Parametrization of the Serial Robot Manipulator

To enable efficient performance evaluation of robotic manipulator concepts, a parameterized and generalizable model must be developed to support simulations and performance analysis. The model proposed in this work builds upon and extends the structural framework of Peter Corke's Python Robotics Toolbox [125]. It represents a serial manipulator as a sequence of interconnected links, where each link comprises a shell and a joint located at its distal end. Joints may be either fixed or revolute, with the latter being actuated. The model incorporates kinematic, structural, and dynamic representations, along with a detailed model of actuator dynamics and limitations. Its primary objective is to provide a flexible modeling framework capable of representing a broad range of serial manipulator architectures, thereby enabling engineers to systematically explore and assess various conceptual designs. The overall structure of the model is illustrated in Figure 3.1.

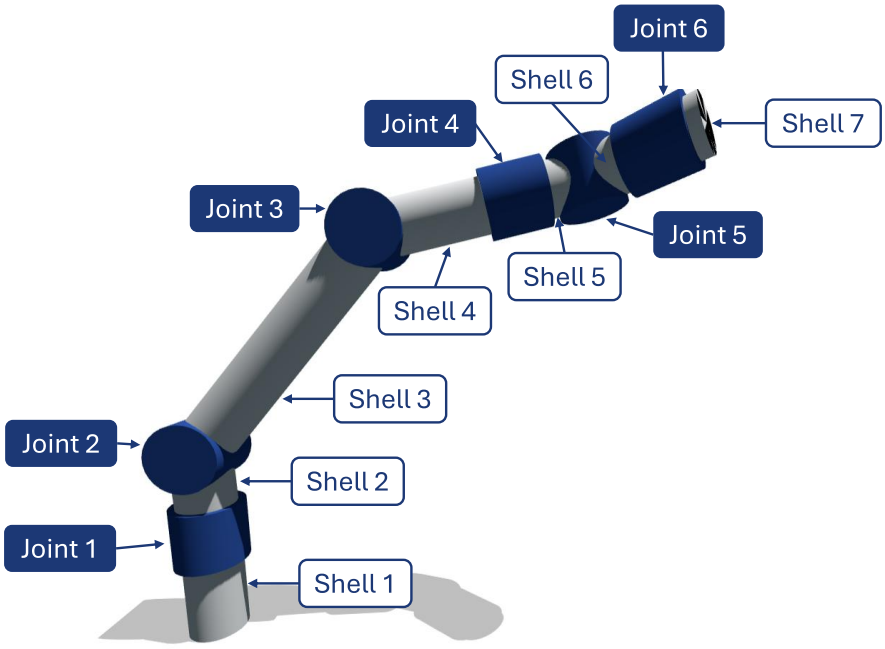


Figure 3.1 Representation of the sequence of links with their respective shells and joints for a 6-DoF manipulator model.

3.1.1 Kinematic Model

The kinematic model of the robot is derived using Elementary Transform Sequences (ETS) [21]. Given that this study focuses exclusively on serial robots, the kinematic model describing the pose of the end-effector is described as the product of the ETS representations of each individual link. Kinematic frames are assigned at the proximal end of each link, and the corresponding ETS defines the HTM from the link's base to its distal end.

The ETS description decomposes the 4×4 HTM into a structured sequence of elementary transformations. For each link, this sequence includes translations along the base frame axes (T_x, T_y, T_z), followed by intrinsic rotations about each axis (R_x, R_y, R_z), as illustrated in Figure 3.2. If a joint is located at the end of the link, the ETS is further extended by an elementary rotation about the Z-axis parameterized by the joint angle q, represented as $R_z(q)$. This convention is consistent with the DH parameterization, in which joint axes are aligned with the Z-axis by definition.

Accordingly, the kinematic parameters associated with each link consist of the six static components of the ETS and an additional parameter indicating the presence of a joint at the distal end of the link, along with two parameters defining the joint's lower and upper motion limits.

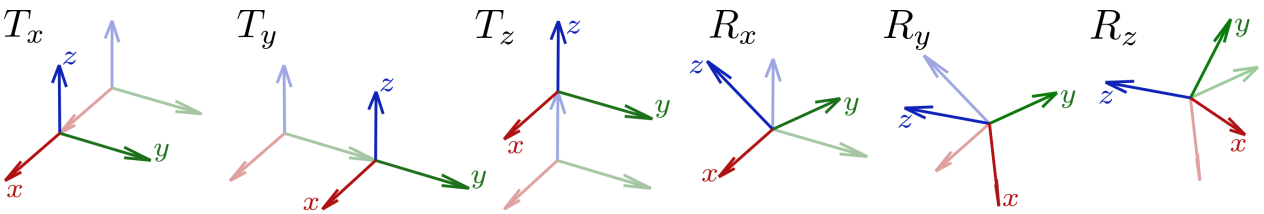


Figure 3.2 Illustration of the elementary transform components within an Elementary Transform Sequence. Translational motions along the x, y and z axes (T_x, T_y, T_z) are followed by intrinsic rotations applied sequentially about the rotated axes (R_x, R_y, R_z).

The complete kinematic model of an n-DoF serial manipulator is thus represented as a chain of elementary transformations with n variable terms and $9n$ parameters. Forward kinematics are evaluated by substituting the variable joint angles q_i into the global ETS and computing the resulting HTM.

Optimized Inverse Kinematics Model

In Chapter 2, the concept of inverse kinematics was introduced, along with the rationale for selecting the LM algorithm. While this algorithm is known for its robust performance and rapid convergence, it shares a common limitation with all numerical, non-analytical inverse kinematics methods: it converges to a single solution. In practice, a manipulator may be capable of achieving the same end-effector pose through multiple distinct joint configurations. For redundant manipulators, the set of feasible solutions may even be infinite. In such cases, identifying the optimal configuration among the many possibilities becomes a critical challenge. This challenge is particularly relevant when analyzing manipulator performance at a specific pose, as the evaluation functions presented in Section 3.2 depend on the associated joint configuration. Simply selecting the first solution returned by the LM algorithm may lead to results that are suboptimal or inconsistent from an engineering perspective. Depending on the application, an engineer may wish to prioritize configurations that maximize payload capacity, minimize end-effector deflection, or maintain joint positions safely away from their limits.

To address this issue, the approach proposed in this thesis formulates a constrained optimization problem aimed at improving a selected secondary objective, such as maximizing payload capacity or minimizing joint limit proximity, while enforcing a constraint on pose reachability (Eq. 3.1).

$$\begin{aligned} \min_{\mathbf{q}} \quad & \text{secondary\_objective}(\mathbf{q}) \\ \text{subject to} \quad & \text{e} \leq \text{tol} \\ & \mathbf{q}_{lower} \leq \mathbf{q} \leq \mathbf{q}_{upper} \end{aligned} \tag{3.1}$$

Where e, the error defined as the deviation between the end-effector pose corresponding to $\mathbf{q}$ and the desired target pose, remains below a specified tolerance (tol), and $\mathbf{q_{lower}}$ and $\mathbf{q_{upper}}$ are the joint limits.

To reduce computational cost and avoid convergence to suboptimal local minima, the constrained optimization problem is solved using a multi-start strategy based on SQP, a gradient-based algorithm (see Section 3.3.4). As summarized in Algorithm 1, n initial joint-space samples $\mathbf{q}_0$ are generated using Latin Hypercube Sampling (LHS) within the joint limits $\mathbf{q}_{lower}$ and $\mathbf{q}_{upper}$. For each sample, the LM inverse kinematics algorithm is used to compute a feasible joint configuration $\mathbf{q}$ that reaches the desired end-effector pose $\mathbf{p}$. If the manipulator is not redundant and at least one solution is found, the procedure terminates and the configuration with the best value for the secondary objective is selected. Otherwise, if the pose is reachable and the robot is redundant, SQP is initialized from each configuration to minimize a given secondary objective function while maintaining pose reachability. This yields a set of locally optimized configurations $\mathbf{q}_{opt,k}$ with associated objective values $f_{opt,k}$. The configuration with the best objective value is then selected as the optimal solution $\mathbf{q}_{opt}$.

Algorithm 1 Optimize Inverse Kinematics

Input: $\mathbf{p}$, secondary_objective
Output: Optimal configuration $\mathbf{q}_{opt}$

```
initial_samples ← LHS(n, q_lower, q_upper)
for each q_0 in initial_samples do
    q ← IK(p, q_0)
    if reachable then
        if not redundant then
            Store q_opt,k = q
            Store f_opt,k=secondary_objective(q)
        end if
        if redundant then
            Store q_opt,k, f_opt,k = SQP(secondary_objective, p, q)
        end if
    end if
end for
q_opt ← best(q_opt,k)
return q_opt
```

Although this method provides excellent results, it does not guarantee finding the global optimal configuration as it depends on the random starting points used at the first step. Also, the step used to assess the numerical approximation of gradient of the secondary function and the termination criteria must be selected appropriately for the algorithm to converge efficiently.

3.1.2 Structural Model

The shell of each link is modeled by a hollow cylinder approximation illustrated in Figure 3.3. The direction and the length l_s of the cylinder is defined by the translational component of the ETS of the child link. Consequently, the structural model of each shell is parametrized by a wall thickness t_s, an outer diameter D_s, and a material density ρ_s.

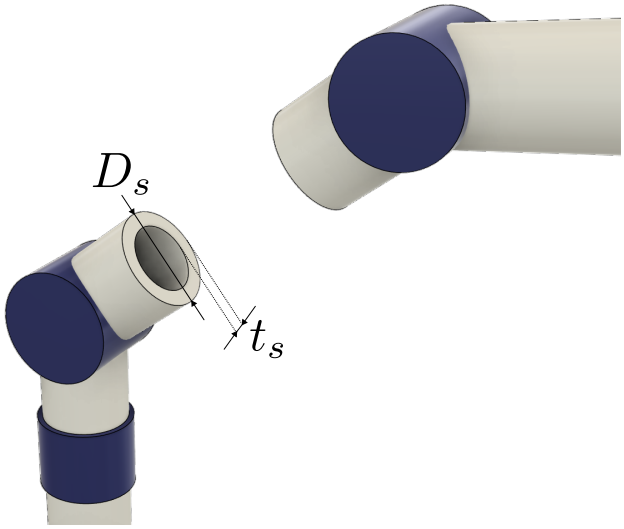


Figure 3.3 Illustration of the model of the link shell and corresponding structural parameters

3.1.3 Actuator Model

A large portion of robotic performance criteria depends directly on the selection of actuators at each joint. For this reason, a versatile design exploration tool must incorporate a complete and flexible actuator model.

A rotary actuator is a complex mechatronic system composed of several integrated components. The two primary elements responsible for actuation are the motor and the reducer. The full assembly may also include an outer shell, driver, encoder, torque sensor, brake, and other additional components. However, for the purposes of this thesis, the actuator model is simplified to focus solely on a motor and a reducer. Specifically, the modeled components

are a direct current (DC) motor and a harmonic reducer, a common combination in robotic systems. This simplification is adopted to reduce modeling complexity, though the framework remains open to future extension with additional motor and reducer types to broaden its applicability. To account for the mass of omitted components, the total actuator mass is approximated considering the combined mass of the motor and reducer to represent 30% of the total mass.

Motor Model

Theoretically, a DC motor exhibits a linear speed–torque relationship. In practice, this relationship is bounded by a plateau corresponding to the maximum torque that the motor cannot exceed. This behavior can be described using three parameters: the maximum torque τ_{m-peak}, the maximum speed at zero torque ω_{m-max}, and the critical speed ω_{m-c}, where the linear relationship reaches the maximum torque limit. Due to thermal and durability considerations, motor manufacturers also specify a maximum nominal torque τ_{m-nom}, which represents the maximum average torque the motor can sustain over a typical operating cycle without degrading its components. Based on these characteristics, three motor performance models are derived for different evaluation purposes:

1. The peak performance model represents the maximum achievable performance of the motor, constrained by the linear torque–speed relationship and the maximum torque limit.

2. The conservative peak performance model provides an approximation of the motor's capabilities, reducing the relationship to two limiting values: maximum torque and critical speed. This model is used when a full torque–speed dependency cannot be considered (see Section 3.2.1).

3. The nominal model is similar to the conservative peak model but applies the maximum nominal torque as the upper torque limit. It is primarily used for assessing actuator performance under static conditions (zero speed and acceleration).

The three different models are illustrated in Figure 3.4, highlighting the model-dependent relationship describing maximum motor torque as a function of speed $\tau_{m-max}(\omega)$.

The motor is thus fully characterized by its mass m_m, rotor inertia J_m, and the four parameters defining its torque–speed relationship.

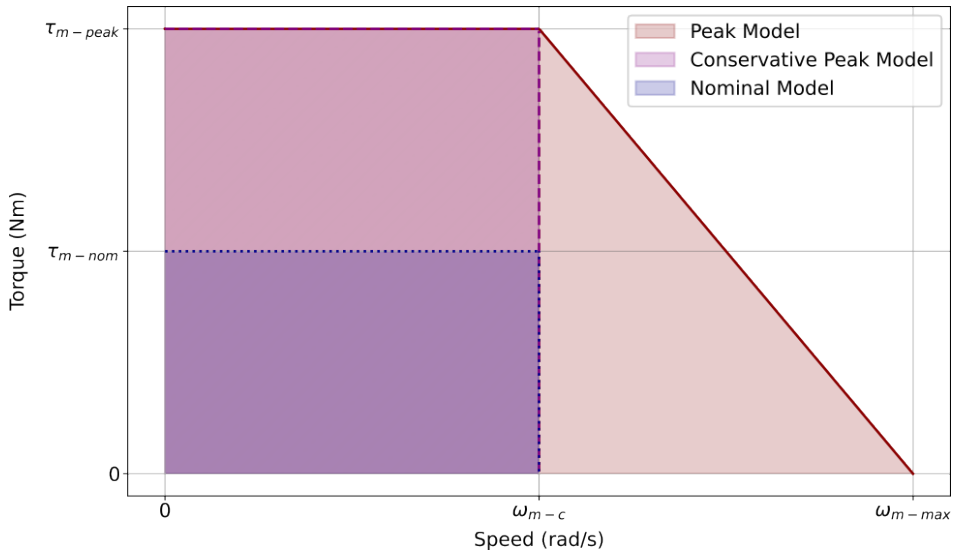


Figure 3.4 Torque-speed relationship for the peak, conservative peak and nominal performance models of a DC motor.

Reducer Model

The torque capacity of a motor is typically insufficient to meet the demands of robotic applications. To address this, a harmonic gearbox is integrated at the motor output, providing torque amplification and speed reduction, characterized by a reduction ratio N and an efficiency η. Manufacturers specify operational limits to ensure durability and prevent damage, such as a maximum output torque τ_{r-max} and a maximum input speed ω_{r-max}. The harmonic reducer is fully characterized by its mass m_r, output shaft inertia J_r, and efficiency η. Additionally, its torsional stiffness k must be considered for stiffness analysis.

Finally, the impact of the inertia of the rotating components must be considered. The dynamics of the actuator, expressing the motor torque and speed as a function of output shaft acceleration and torque, are detailed in Equation 3.2:

$$\begin{aligned} \tau_m &= (J_m + \frac{J_r}{N^2}) * N\ddot{q} + \frac{\tau_{output}}{\eta N} \\ \omega_m &= \frac{\omega_{output}}{N} \end{aligned} \tag{3.2}$$

where τ_m and ω_m are the torque and speed produced by the motor, $\ddot{q}$ is the angular acceleration at the actuator output, and τ_{output} and ω_{output} are the torque and speed at the output shaft. Frictional forces are neglected in this model for simplification purposes.

The derived actuator model can be used in two different ways. First, it can be used to verify

whether a given combination of output torque τ_{out}, speed ω_{out}, and acceleration $\ddot{q}$ at the output shaft is achievable. To do so, the necessary motor torque and speed and assessed with the actuator dynamics (Eq. 3.2) and the results are compared with the motor and reducer torque and speed limits. The constraints for feasibility are detailed in the following equations.

$$\tau_{out} \leq \tau_{r-max} \tag{3.3}$$

$$\omega_m \leq \omega_{r-max} \tag{3.4}$$

$$\omega_m \leq \begin{cases} \omega_{m-max} & \text{if peak model} \\ \omega_{m-c} & \text{otherwise} \end{cases} \tag{3.5}$$

$$\tau_m \leq \tau_{m-max}(\omega_m) \tag{3.6}$$

For the conservative peak model, the aim is to provide state independent bounds, therefore, the fact that the maximum torque is dependent of acceleration is an issue. The proposed solution is to neglect the impact of acceleration in the dynamic model. For the acceleration component to be negligible with respect to the output torque component, it should be inferior with a ratio of at least 10. This ratio can be enforced by adding an acceleration limit defined by the following equation:

$$\ddot{\mathbf{q}}_{max} = \frac{N * \tau_{m-max}}{10(J_m * N^2 + J_r)} \tag{3.7}$$

The model can also be used in a different way, assessing the maximum torque the actuator is capable of producing given a combination of speed ω_{out} and acceleration $\ddot{q}$ at the output shaft. It is particularly useful for payload capacity evaluation (see Section 3.2.1). Similarly to the previous use, the motor speed is assessed using the actuator dynamics (Eq. 3.2). At this stage, if the motor speed exceeds motor and reducer limits, the returned torque capacity is null. Otherwise, the maximum motor torque is calculated using the chosen motor model $\tau_{m-max}(\omega_m)$ and the dynamics equations are applied inversely to extract the maximum output torque due to motor limits $\tau_{out-max-motor}$.

Finally, the maximum output torque of the actuator is deduced as the minimum between the calculated maximum output torque due to motor limits $\tau_{out-max-motor}$ and the reducer maximum output torque τ_{r-max}.

These two applications of the actuator model provide the necessary tools to evaluate the

performance of a manipulator design concept on actuator-dependent criteria.

3.1.4 Dynamic Model

The general dynamic model of the manipulator has already been introduced in Section 2.1.2. The RNE algorithm only needs information on the inertial parameters of each link to establish the general inertia matrix and compute the joint torques necessary for a dynamic motion. The inertial parameters of a link are its mass, its center of mass and its matrix of inertia, this section presents a rapid overview of the calculations needed to extract these parameters from shell and actuator properties.

First of all, the mass of the link is computed as the sum of the mass of the shell m_{shell} plus, for links with actuators, the mass of the actuator at the end of the link m_{act} (Eq.3.8).

$$\begin{aligned} m_{shell} &= \rho_s \pi l_s t_s (D_s - t_s) \\ m_{link} &= m_{shell} + m_{act} \end{aligned} \tag{3.8}$$

The computation of the center of mass is determined considering the actuator as a point mass located at the origin of the distal frame of the actuator. For a link where the translational component of the ETS from the proximal frame to the distal frame is $\mathbf{t} = \begin{bmatrix} T_x & T_y & T_z \end{bmatrix}^T$, the center of mass $\mathbf{r}$ of the link is expressed in Equation 3.9.

$$\mathbf{r} = \frac{m_{shell} * \frac{1}{2}\mathbf{t} + m_{act} * \mathbf{t}}{m_{tot}} \tag{3.9}$$

The inertia matrix, expressed with respect to the center of mass of the link, is computed with Equation 3.10.

$$\mathbf{I_{link}} = \mathbf{I_{shell}} + \mathbf{I_{actuator}} \tag{3.10}$$

Where $\mathbf{I_{shell}}$ is has been rotated and translated to the appropriate frame and $\mathbf{I_{actuator}}$ is the inertia matrix of a point mass m_{act} at the distal end of the actuator.

3.1.5 Overview of the Manipulator Model

To summarize, the manipulator is modeled by a series of links and actuators. Each link can be fully represented by kinematic and structural design parameters. The actuators, present at joints between links are modeled with a motor and a reducer and their corresponding physical properties and dynamic limits. Tables 3.1 and 3.2 present the complete set of design

parameters needed to fully model a link and an actuator respectively. An example of a complete parametrization of the model for the Kinova GEN3 manipulator [30] is provided in Appendix A.

Table 3.1 Summary of the design parameters for a manipulator link model and corresponding symbol and units.

Parameter	**Symbol and Unit**
Translational Component X of ETS	T_x (m)
Translational Component Y of ETS	T_y (m)
Translational Component Z of ETS	T_z (m)
Rotation about X-axis of ETS	R_x (rad)
Rotation about Y-axis of ETS	R_y (rad)
Rotation about Z-axis of ETS	R_z (rad)
Presence of a Joint at the Distal End	`has_joint` (`True` or `False`)
Joint Limit (Minimum)	q_{min} (rad)[1]
Joint Limit (Maximum)	q_{max} (rad)[1]
Shell Wall Thickness	t_s (m)
Shell Outer Diameter	D_s (m)
Shell Material Density	ρ_s (kg/m^3)

[1] Only applicable if `has_joint` is `True`.

Table 3.2 Summary of the design parameters for a actuator model and corresponding symbol and unit.

Parameter	**Symbol and Unit**
Motor Mass	m_m (kg)
Motor Rotor Inertia	J_m (kg $\cdot$ m^2)
Motor Maximum Peak Torque	τ_{m-peak} (Nm)
Motor Maximum Nominal Torque	τ_{m-nom} (Nm)
Motor Maximum Speed	ω_{m-max} (rad/s)
Motor Critical Speed	ω_{m-c} (rad/s)
Reduction Ratio	N
Gear Efficiency	η
Reducer Mass	m_r (kg)
Reducer Output Inertia	J_r (kg $\cdot$ m^2)
Reducer Maximum Output Torque	τ_{r-peak} (Nm)
Reducer Maximum Input Speed	ω_{r-max} (rad/s)
Reducer Torsional Stiffness	k (Nm/rad)

3.2 Performance Criteria and Evaluation Functions

To ensure versatility, the tool requires a diverse set of evaluation functions that can assess various performance criteria. As presented in Chapter 2, a large variety of performance criteria exist to evaluate manipulator performance. Although any scalar performance indicator could be implemented with the presented methodology, the evaluation functions implemented in this project are selected based on the relevance of the performance criteria. To identify performance criteria that are both theoretically grounded and practically relevant, a survey of the scientific literature and commercial data-sheets was conducted across 77 articles and 15 data-sheets. This dual-source approach ensures a comprehensive understanding of current trends and priorities used by both researchers and industry practitioners. The following section presents the functions developed to evaluate the most frequently mentioned indicators, differentiating task related criteria from global performance indicators. Given the diversity of potential designs, these functions are designed to accommodate any combination of design parameters.

3.2.1 Task-Specific Criteria

The selection of appropriate task related performance criteria is typically guided by the nature of the task for which the robot is intended. In this project, tasks are categorized into three hierarchical classes:

1. Positioning-based tasks – defined by a set of discrete points or poses that the robot must reach, without a prescribed order or timing.
2. Path-based tasks – defined by an ordered sequence of poses that the robot must follow, where continuity of motion between poses is essential.
3. Trajectory-based tasks – defined by a sequence of poses, parametrized with time intervals, where both the order and timing of the motion are critical.

Performance criteria applicable to positioning-based tasks generally remain relevant for path- and trajectory-based tasks. Additional task-specific criteria can be evaluated to characterize the performance of a robot on a path or trajectory.

Position-Based Tasks

Position-based tasks are characterized by a set of points or poses that the robot must attain to perform a task, for example a set of drilling poses [11] or a set of points describing the

ideal workspace of an assistive robot [42].

Since these points or poses are neither ordered nor associated with specific time steps, the performance evaluation for this task type is conducted independently at each point or pose. The result is a mapping of the performance of the robot over the task. Global task-level performance can then be summarized using statistical measures such as the worst, best, or average value of a selected performance indicator.

Among the literature surveyed, reachability emerged as the most commonly used and fundamental performance indicator for position-based tasks, appearing in 35% of the reviewed studies. Reachability must be verified before any other performance metric can be assessed at a given point or pose, as it determines whether a valid joint configuration exists that allows the robot to reach the target.

The reachability evaluation function thus serves two critical purposes: (1) to compute the reachability indicator for the task, and (2) to determine the joint configurations required to reach each target point or pose. This latter role is essential, as it effectively maps the task into the joint space, where other performance indicators are evaluated.

As presented in Chapter 2, the reachability of each point or pose in the task can be evaluated with an inverse kinematics algorithm. In the context of this framework, a key feature is the use of the optimized IK algorithm (IK_{opt}) introduced in Section 3.1.1, which incorporates a secondary objective, here arbitrarily selected as joint-limit avoidance. The general procedure for reachability evaluation is illustrated in Figure 3.5.

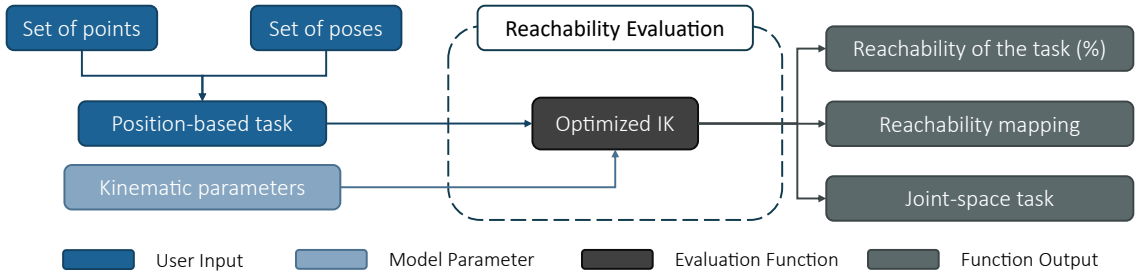


Figure 3.5 Reachability evaluation workflow illustrating how user-defined tasks and kinematic model parameters are processed to generate the reachability metric, reachability mapping, and joint-space task representation.

Solving the IK problem for each point or pose in the task generates the corresponding joint-space representation, denoted as $\mathcal{Q}$. The definition of the reachability indicator itself depends on the context in which the function is evaluated. When reachability is used within the inverse design process, as an objective function or as a constraint ensuring full task coverage, it may

be desirable for the function to exhibit continuity, allowing the optimization algorithm to more effectively explore the design space. In such cases, the indicator can be defined as the sum of IK errors across all task points or poses. Conversely, in the context of the direct design process, or when reachability is treated as an inequality constraint with a target threshold, a more interpretable metric is preferable. In this case, the reachability indicator described in Section 2.2.1, which represents the ratio of reachable points or poses, should be used. The complete procedure for evaluating reachability is outlined in Algorithm 2.

Algorithm 2 Assess Task Reachability

Input: $\mathcal{P}$: list of N points/poses
Output: Reachability (%), Sum of errors (Σ_{errors}), Joint space task ($\mathcal{Q}$)

```
N_reachable ← 0
Q ← ∅
Σ_errors ← 0
for each pose in P do
    q, error ← IK_opt(pose)
    if error < tolerance then
        N_reachable ← N_reachable + 1
        Q ← Q ∪ {q}
    end if
    Σ_errors ← Σ_errors + error
end for
Reachability ← (N_reachable / N) × 100
return Reachability, Σ_errors, Q
```

The remaining performance criteria are evaluated using the set of joint configurations $\mathcal{Q}$, which corresponds to the solutions obtained for each task point or pose. Among the criteria found in the surveyed literature and commercial data-sheets, the most prevalent for position-based tasks are payload capacity, featured in all data-sheets reviewed, and manipulability, end-effector deflection, dexterity and natural frequency, which appeared in 16%, 9%, 5% and 5% of the literature sources, respectively. The evaluation functions for natural frequency, manipulability and dexterity are directly implemented from their definition in Section 2.2. The implementation of payload capacity and end effector deflection evaluations are detailed in the following paragraphs. The general process for their evaluation on position-based tasks is illustrated in Figure 3.6.

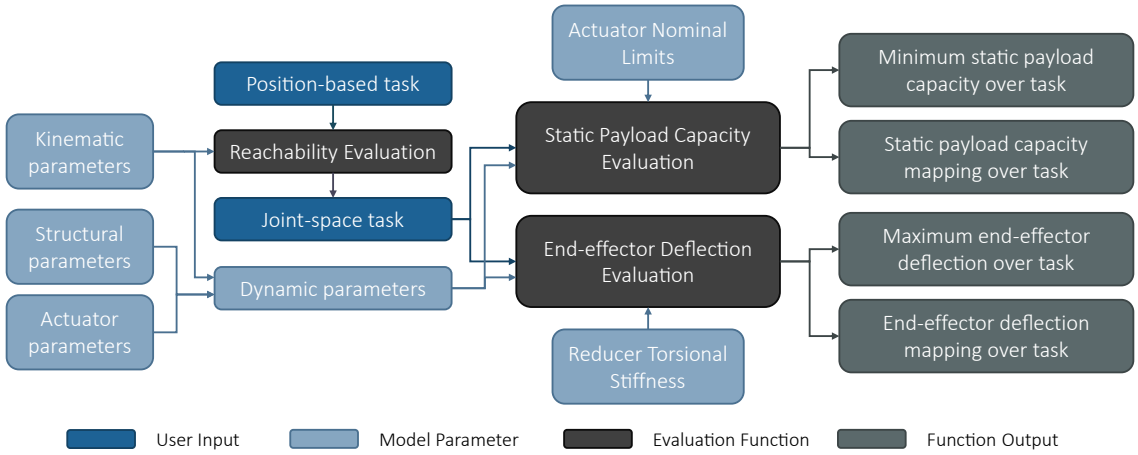


Figure 3.6 Static payload capacity and end-effector deflection evaluation workflow illustrating how user-defined tasks and model parameters are processed to generate the performance metrics and mappings.

The static payload capacity of a robot in a given configuration $\mathbf{q}$ is defined as the maximum payload mass that can be supported at the end effector. The evaluation process is divided into the following steps:

1. Actuator Torque Limits: The maximum static torque capacity of each actuator, denoted $\boldsymbol{\tau}_{\max}$, is calculated based on the nominal model of the actuators.

2. Gravity Compensation: The torques required to maintain the robot's configuration without any payload, $\boldsymbol{\tau}_{\text{gravity}}$, are computed using the RNE algorithm with zero velocities and accelerations.

3. Payload-Induced Torques: The joint torques needed to compensate for a 1-kg payload placed at the end effector in the presence of gravity vector $\mathbf{g}$, denoted $\boldsymbol{\tau}_{\text{payload}}$, are calculated using the Jacobian. $\boldsymbol{\tau}_{payload} = -\mathbf{J}^T.\mathbf{g}$

4. Maximum Load Estimation: The total torque required to support an arbitrary payload is modelled as the sum of gravity compensation torques and a scaled version of the payload-induced torques. The maximum payload capacity corresponds to the smallest positive scaling factor k such that the total torque of a joint reaches its respective actuator limit. If the minimum k is negative, it indicates that the robot is unable to support its own weight in that configuration.

Allowing the payload capacity indicator to take negative values facilitates more effective exploration of the design space in the inverse design process. Physically, a negative payload capacity corresponds to the upward force (in kgf) that must be applied at the end-effector to compensate for gravitational torque, ensuring that the torque required at the limiting joint

does not exceed the torque limits of the actuator.

The complete evaluation process is presented in Algorithm 3.

Algorithm 3 Compute Maximum Static Payload Capacity

Input: $\mathbf{q}$
Output: Maximum payload capacity (kg)
$\boldsymbol{\tau}_{gravity} = RNE(\mathbf{q}, \mathbf{0}, \mathbf{0})$
$\boldsymbol{\tau}_{payload} = -\mathbf{J}^T.\mathbf{g}$
Solve for $\mathbf{k}$: $|\boldsymbol{\tau}_{gravity} + \mathbf{k} * \boldsymbol{\tau}_{payload}| = \boldsymbol{\tau}_{max}$
return Maximum payload capacity = $\min(\mathbf{k})$

The translational deflection of the end effector in a given robot configuration $\mathbf{q}$ is estimated using the joint stiffness matrix, as described in Section 2.2.3. The evaluation process proceeds as follows:

First, the joint-space deflection is calculated by multiplying the gravity-induced torque vector by the inverse of the joint stiffness matrix.

Next, a linear approximation of the end-effector's Cartesian deflection is obtained by multiplying the Jacobian matrix by the joint-space deflection vector. The translational deflection scalar is then computed as the Euclidean norm of the translational components of the resulting Cartesian deflection vector.

The full procedure for evaluating end-effector translational deflection is outlined in Algorithm 4.

Algorithm 4 Compute End-Effector Deflection

Input: $\mathbf{q}$
Output: End-effector deflection (mm)
$\boldsymbol{\tau}_{gravity} = RNE(\mathbf{q}, \mathbf{0}, \mathbf{0})$
$\Delta\boldsymbol{\theta} = \boldsymbol{K}_{\theta}^{-1}\boldsymbol{\tau}_{gravity}$
$\Delta\mathbf{p} = \mathbf{J}.\Delta\mathbf{q}$
$\Delta\mathbf{t}$ = extract_translational_components($\Delta\mathbf{p}$)
return $\|\Delta\mathbf{t}\|_2$

All of the performance criteria presented are inherently local, meaning they are evaluated at individual configurations corresponding to specific task points or poses. As a result, the analysis outcomes can be presented in two ways: either as a spatial mapping, where a color map visualizes the criterion's value at each task point or pose, or as a derived scalar indicator, summarizing the robot's performance over the entire task.

For instance, metrics such as the minimum static payload capacity, the average manipulability, or the maximum end-effector deflection across the task can be reported. An example of

such a spatial representation, specifically, the static payload capacity evaluated over a task defined by uniformly distributed points within a cube, is shown in Figure 3.7.

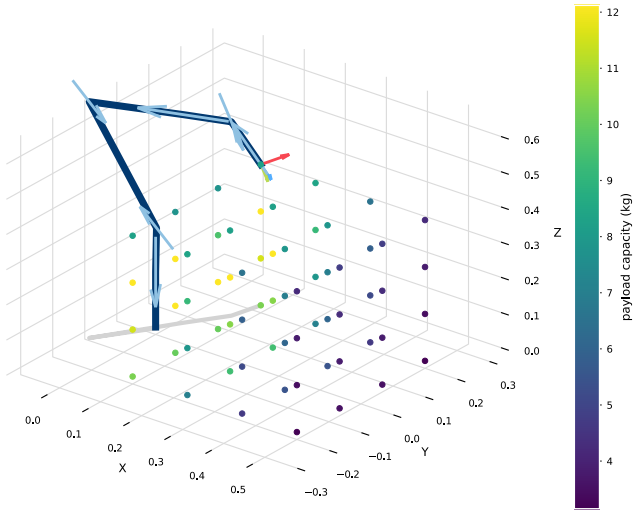


Figure 3.7 Mapping of the static payload capacity of a robot manipulator (represented with a wireframe model) over a task composed of uniformly spaced out points within a cube.

Path-Based Tasks

Path-based tasks are a subcategory of position-based tasks in which the order of the task poses is crucial, as the robot is required to follow a continuous path that passes through each specified point. This category is particularly relevant in applications involving repetitive and cyclic motions, such as pick-and-place operations.

Before any performance indicators can be evaluated for a path-based task, the reachability of all task poses must first be verified. If the entire path is deemed reachable, a corresponding joint-space path must be generated to guide the end-effector through the specified Cartesian poses. To ensure path smoothness, the optimized inverse kinematics algorithm (IK_{opt}) is used to determine the initial configuration for the first pose, selecting the solution that maximizes distance from joint limits. For the remaining poses, the inverse kinematics is initialized using the solution of the preceding pose, promoting continuity in joint space. A smooth joint-space path is then generated via cubic spline interpolation between successive configurations.

All performance criteria previously introduced for position-based tasks can also be evaluated at each configuration along the path.

Among the performance indicators relevant to path-based tasks, minimum cycle time, the shortest possible time for the robot to execute the full trajectory, is the most widely used.

It appears in 2 of the 15 surveyed data-sheets and in 13% of the reviewed literature. The problem of determining the minimum execution time along a known path is known as a Time Optimal Path Parametrization (TOPP) problem. The general process for minimum cycle time evaluation is illustrated in Figure 3.8.

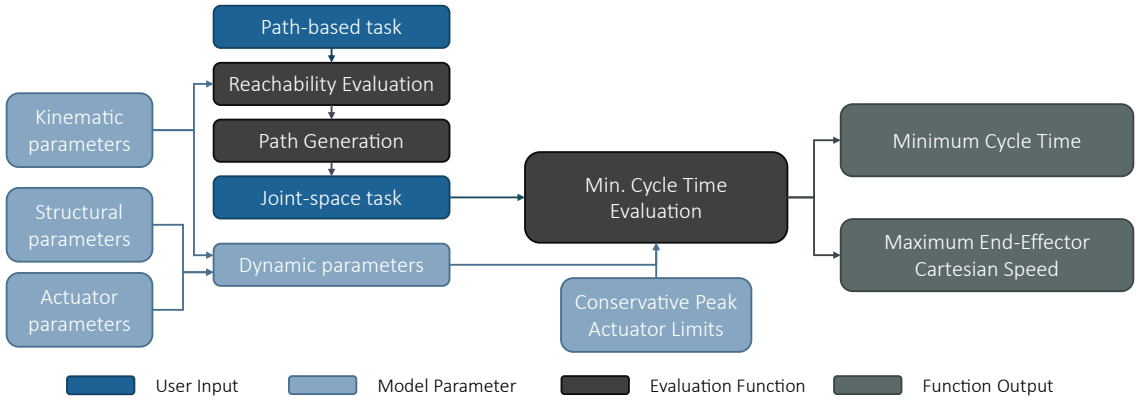


Figure 3.8 Evaluation workflow for the determination of the minimum cycle time necessary for a manipulator to execute a path, illustrating how user-defined tasks and model parameters are processed to evaluate the performance of a manipulator on a path-based task.

The current state-of-the-art solution for this problem is TOPP-Reachability Analysis (TOPP-RA) [126], which is favoured due to its computational efficiency and high success rate. In the context of a feasible path, the primary constraints that limit reductions in cycle time are the actuator limits introduced in Section 3.1.3. However, since TOPP-RA is only compatible with state-independent constraints, it cannot take into account the torque-speed relationship of the peak motor performance model. As a result, the algorithm is executed using the conservative peak model of each actuator, which provides constant torque, speed, and acceleration bounds.

The output of this evaluation is the minimum cycle time. In addition, the resulting joint-space trajectory can be mapped to Cartesian space via forward kinematics, allowing for the computation of the maximum Cartesian speed along the path, another performance criterion that appears in 60% of the surveyed data-sheets.

Trajectory-Based Tasks

Trajectory-based tasks are a specialized subclass of path-based tasks in which the path is explicitly parameterized with time steps. For instance, an industrial robot operating along a conveyor belt may need to reach pick-and-place positions at specific time instants.

As with position- and path-based tasks, the first step in analyzing a trajectory-based task is to convert the Cartesian task into a joint-space representation. This process is identical to that

used for path-based tasks, with the exception that the cubic spline interpolation incorporates the time parametrization provided in the task definition. The resulting joint-space trajectory thus contains time-dependent joint positions, velocities, and accelerations.

The most frequently encountered performance indicators for trajectory-based tasks in the surveyed sources are trajectory feasibility (31% of the literature) and power- and energy-related criteria (5% of the literature, 66% of data-sheets).

Trajectory feasibility is evaluated by checking whether the actuators can produce the required joint torques and velocities at each point along the trajectory. This is done by computing the torque profile using the Recursive Newton-Euler (RNE) algorithm and verifying that the torques and velocities fall within the bounds of the actuator's peak performance model. Since this criterion is binary, the trajectory is either feasible or not, a more informative metric called dynamic payload capacity is introduced to quantify feasibility in a continuous and interpretable manner. The general process for its evaluation is illustrated in Figure 3.9.

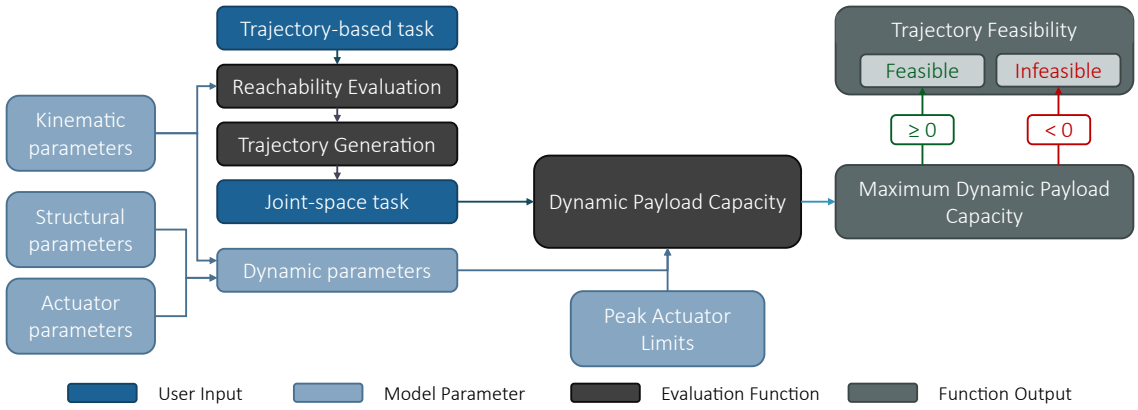


Figure 3.9 Dynamic payload capacity evaluation workflow illustrating how a user-defined task and model parameters are processed to assess trajectory feasibility and determine the maximum payload the manipulator can handle during execution.

The dynamic payload capacity at a given trajectory step represents the maximum mass that can be held at the end effector while satisfying all dynamic constraints. Its evaluation differs from the static case due to the inclusion of inertial effects and joint accelerations, which render the static force/torque relationship invalid. The evaluation procedure is as follows:

1. Actuator Torque Limits: The joint torque limits are computed using the actuator peak model, taking joint velocities into account.

2. Baseline Torques: The baseline joint torques ($\boldsymbol{\tau}_0$) required to achieve the desired motion and resist gravity and inertial effects are computed using the RNE algorithm with no payload.

3. Payload-Induced Torques: The procedure is repeated by adding a point mass of one kilogram at the end effector. The additional torques caused by the payload ($\boldsymbol{\tau}_{payload}$) are derived as the difference from the baseline torques.

4. Maximum Load Estimation: Given the linearity of torque response to added payload for a fixed state, a scaling factor is derived to determine the maximum payload the robot can carry before any joint exceeds its torque limit.

The complete evaluation is detailed in Algorithm 5. The overall dynamic payload capacity of a robot over a trajectory is defined as the minimum capacity computed across all steps of the trajectory.

Algorithm 5 Compute Maximum Dynamic Payload

Input: $\mathbf{q}$, $\mathbf{q}$', $\mathbf{q}$"
Output: Maximum dynamic payload (kg)
Evaluate joint torque limits $\boldsymbol{\tau}_{\text{peak}} = \text{peak\_performance\_model}(\mathbf{q}')$
$\boldsymbol{\tau}_0 = RNE(\mathbf{q}, \mathbf{q}', \mathbf{q}'')$
Add 1 kg point mass to end effector frame
$\boldsymbol{\tau}_{payload} = RNE(\mathbf{q}, \mathbf{q}', \mathbf{q}'') - \boldsymbol{\tau}_0$
Solve for $\mathbf{k}$: $|\boldsymbol{\tau}_0 + \mathbf{k} * \boldsymbol{\tau}_{payload}| = \boldsymbol{\tau}_{peak}$
return min($\mathbf{k}$)

Joint space trajectories can be evaluated for all the previously described position-based performance criteria.[1]

For the purpose of this project, the power and energy related criteria implemented in the framework focus on mechanical power and work. The mechanical power consumption over the trajectory is evaluated as the product of torque and speed on the output shaft of the motor. Average and peak power consumption can be evaluated considering the sum of each actuator's power consumption. The mechanical work is evaluated by integrating instantaneous power over the trajectory. While straightforward to compute, these indicators are effective tools for assessing the energy efficiency of a robot.

3.2.2 Global Criteria

Although the previous evaluation functions are very relevant when considering the performance of a robot on a specific task, indicators describing the global performance of the robot must also be implemented. This section presents the two classes of evaluation functions for

[1]The analysis of end effector deflection for trajectory-based tasks requires the additional consideration of joint speeds and accelerations when computing joint torques.

global criteria implemented in the framework; optimization-based techniques and workspace analysis based techniques.

Optimization Techniques for Global Criteria

One method to derive a global performance indicator from a local criteria is to identify the joint configuration that yields the worst-case or best-case value of that indicator. For instance, determining the configuration with the lowest static payload capacity defines the robot's overall static payload capacity, while identifying the configuration with the greatest end-effector reach defines the robot's "maximum reach". These two global indicators are consistently reported in the examined data-sheets. Identifying such extreme configurations can be achieved either through exhaustive sampling of the joint space or through optimization-based methods.

For this project, it was decided to leverage optimization techniques in order to reduce computation time and increase precision. Specifically, the SQP algorithm is used to find the maximum reach, minimum static payload capacity, maximum end effector deflection and minimum natural frequency over the workspace of the robot. It must be emphasized that the optimization techniques presented here are not part of the inverse design procedure but a method to efficiently compute global criteria, the optimized variables here are joint angles and not design variables. The general procedure used to identify the value of the globalized local indicator is illustrated in Figure 3.10.

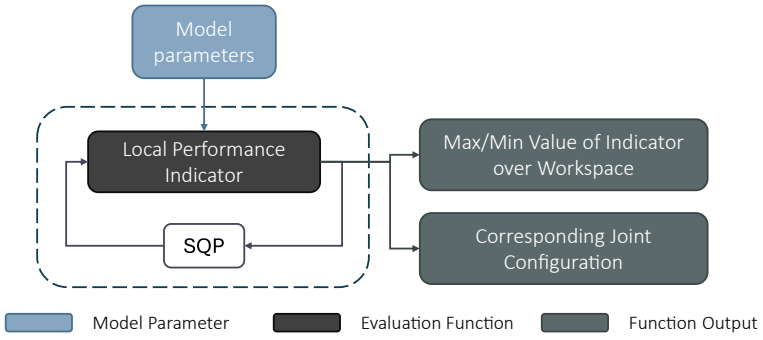


Figure 3.10 General procedure of the identification of critical joint configurations for the globalization of local manipulator performance criteria using optimization techniques.

The optimization problems follow a common structure: a local performance indicator is optimized while ensuring that the joint configuration $\mathbf{q}$ remains within the specified joint limits ($\left[\mathbf{q}_{\min}, \mathbf{q}_{\max}\right]$). For example, the formulation for maximizing reach in a given direction $\mathbf{d}$ is shown in Equation 3.11, where $\mathbf{p}(\mathbf{q})$ denotes the end-effector position resulting from

forward kinematics:

$$\begin{aligned} \max_{\mathbf{q}} \quad & \mathbf{d}^T \mathbf{p}(\mathbf{q}) \\ \text{subject to} \quad & \mathbf{q}_{\min} \leq \mathbf{q} \leq \mathbf{q}_{\max} \end{aligned} \tag{3.11}$$

It should be noted that the objective functions for these local indicators are generally non-convex, and thus may contain multiple local optima. To mitigate the risk of convergence to suboptimal solutions, a multi-start strategy is used. Initial guesses for the optimizer are generated using LHS across the joint space, increasing the likelihood of identifying a global optimum.

Reachable Workspace-Based Global Criteria

The second method implemented in the framework for assessing global performance indicators involves modeling the robot's reachable workspace as a position-based task composed of discrete Cartesian points. The general process of this method is illustrated in Figure 3.11. The reachable workspace evaluation relies on the sampling-based method described in Section 2.2.1.

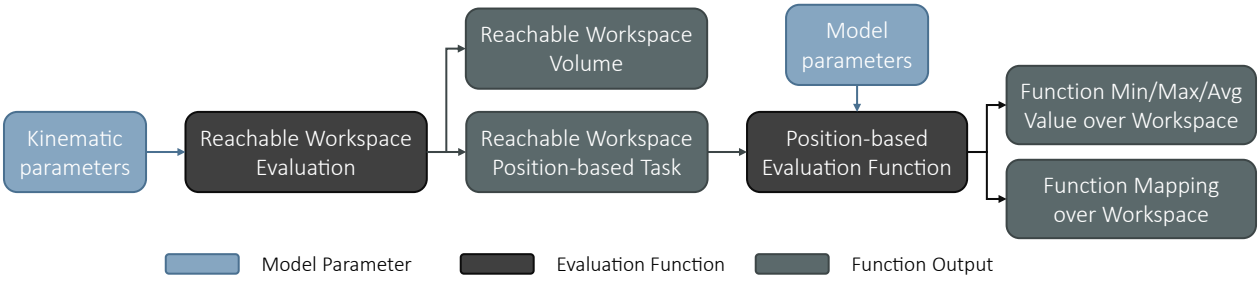


Figure 3.11 Process of global criteria evaluation using the reachable workspace as a task

1. Workspace Boundaries: The boundaries of the robot's reachable Cartesian workspace are first estimated by evaluating the maximum reach in the positive and negative directions along each principal axis. These boundaries are used to discretize the Cartesian space into a uniform grid of cubic voxels.

2. Joint Space Sampling: A large number of joint configurations are sampled using LHS, ensuring a representative and evenly distributed set of samples across the joint space.

3. Workspace Mapping: Forward kinematics is applied to each sampled configuration to compute the corresponding end-effector pose. The Cartesian positions are mapped to the discretized voxel grid, where a voxel is marked as "reachable" (assigned a value of 1) if any

sample falls within it. This mapping results in a binary occupancy map of the reachable workspace.

Finally, the volume of the reachable workspace can be obtained by summing the volumes of all voxels marked as reachable.

In certain cases, major simplifications can be applied to significantly reduce the computational load:

1. If the axis of the first joint aligns with the gravity direction and its limits allow full rotation, the workspace exhibits axial symmetry around this axis. The reachable workspace can then be projected as a 2D cross-sectional slice without loss of generality.
2. If the translational components of the ETS of the final link are entirely aligned with the last joint's axis, the last joint does not affect the reachable workspace.

These simplifications, commonly observed in commercial manipulators, allow two joint variables to be excluded from sampling, significantly improving resolution for a fixed number of samples.

Once computed, the reachable workspace can be treated as a position-based task, enabling the evaluation and spatial mapping of the criteria presented in Section 3.2.1 over the workspace.

Finally, two additional global indicators are implemented in the framework: the GCI and the SLI presented in Section 2.2.1, used in 9% and 5% of the surveyed literature, respectively. A comprehensive summary of all evaluation functions implemented in the tool is presented in Table 3.3, detailing for each function: the design parameters (DP) that influence its output, relevant outputs, and the task types to which it applies.

Table 3.3 Summary of implemented performance evaluation functions, associated influencing design parameters - kinematic (K), structural (S), actuators (A).

Function	**DP**	**Output**	**Task Type**
Reachability	K	reachability percentage [%] reachability map	all
Manipulability	K	min. manipulability [unitless] manipulability map	all
Dexterity	K	avg. dexterity index [unitless] dexterity map	all
Natural Frequency	KSA	min. natural frequency [Hz] natural frequency map	all
Static Payload Capacity	KSA	max. payload capacity [kg] payload capacity map	all
End Effector Deflection	KSA	max. e.e. deflection [mm] e.e. deflection map	all
Cycle Time	KSA	min. cycle time [s] max. cartesian speed [m/s]	path
Trajectory Feasibility	KSA	True or False	trajectory
Power Consumption	KSA	avg. power consumption [W] max power consumption [W]	trajectory
Mechanical Work	KSA	total mechanical work [J]	trajectory
Dynamic Payload Capacity	KSA	min. payload capacity [kg]	trajectory
Max. Reach	K	maximum reach [mm]	global
Max. Static Payload Capacity over Workspace	KSA	max. payload capacity [kg]	global
Max. End Effector Deflection over Workspace	KSA	max. e.e. deflection [mm]	global
Min. Natural Frequency over Workspace	KSA	min. natural frequency [Hz]	global
Reachable Workspace Volume	K	reachable workspace volume [m^3] mapping of reachable workspace	global
SLI	K	SLI index [unitless]	global
GCI	K	GCI index [unitless]	global
Mass	KSA	total mass [kg]	global

3.3 Inverse Design Methodology

The model and performance criteria presented in the previous section allow an engineer to evaluate a manipulator concept using the direct design method, by parameterizing the model and evaluating it with a selection of performance evaluation functions. While direct design methodologies follow a direct path, starting from predefined design parameters and evaluating performance accordingly, they often require numerous iterations to converge toward a satisfactory solution. In contrast, the inverse design method inverts this process: performance objectives and constraints are specified upfront, and the design variables are iteratively optimized to meet these criteria. This paradigm shift enables a more targeted exploration of the design space and is particularly useful when performance-driven customization is desired from the outset.

The model and performance evaluation functions form the core of the inverse design framework. The overall process is structured as follows:

1. A robot model is constructed by defining a set of design parameters. A subset of these parameters is selected as design variables, which are allowed to vary within specified bounds during the optimization process.

2. One or more performance criteria are chosen as objective functions to be either maximized or minimized. When multiple objectives are defined, the problem becomes a MOO problem.

3. Additional performance criteria may be specified as constraints, imposing lower or upper bounds on their permissible values.

4. An appropriate optimization algorithm is selected and executed on the defined problem. It iteratively evaluates the objective and constraint functions by calling the corresponding performance evaluation functions, attempting to converge toward an optimal solution.

5. Upon completion, the best-performing design encountered during the optimization is retained. In the case of multi-objective optimization, a set of optimal solutions forming the Pareto front is returned.

The following sections present the framework used to construct and solve the optimization problem based on the models and evaluation functions introduced earlier.

3.3.1 Design Variables Selection

The design variables are selected from the set of design parameters representing the robot model. These design variables are categorized into two types: continuous variables and discrete variables. Continuous variables can take on any value within a user-defined range,

while discrete variables are restricted to specific, predefined values, representing categorical or binary choices.

The components of each link's ETS can be defined as continuous design variables, as can the joint limits for any joint present at the link extremities. The presence or absence of a joint at the distal end of a link is modeled as a discrete design variable, encoded as a binary value (1 for presence, 0 for absence). Similarly, the shell diameter and thickness parameters in the structural model may be selected as continuous variables, while the material type is treated as a discrete variable. However, since the structural strength and its effect on end-effector deflection are not implemented in the framework, it is not recommended to vary these structural parameters during optimization. Similarly, because self-collision is not considered, considering joint limits as design variables may result in physically impossible designs.

The actuator model constitutes another set of potential design variables and can be handled in two distinct ways. In the first approach, a predefined set of complete actuator models is provided, and the optimization algorithm selects from this set using an integer variable to represent each option. This approach is especially suitable for applications where a robotics company has a fixed catalog of actuators. In the second approach, a more granular design is enabled by separately selecting the motor and reducer from component catalogs, allowing for more customized actuator configurations. However, this added flexibility comes with increased design complexity, as it implies the creation of custom actuators.

To enable the use of optimization algorithms that do not natively handle integer or categorical variables, a simple rounding strategy is used. Discrete design variables are allowed to take on continuous values during the optimization, but are rounded to the nearest valid integer when mapped to their corresponding categorical selections. This approach allows for consistent performance comparisons across all implemented optimization algorithms, regardless of their native support for mixed-integer problems.

3.3.2 Objective Function Selection

The objective function(s) for the optimization are selected from the set of performance evaluation functions outlined in Section 3.2. While all the presented functions can be used as objective functions, their characteristics must be carefully considered when formulating the optimization problem and selecting the appropriate algorithm.

One important characteristic to consider is the continuity of the function, which directly affects the choice of optimization algorithm. The trajectory generation capacity function is non continuous by nature, taking only binary value. Similarly, the reachability and dexterity

evaluation functions, when defined as a percentage of reachable or fully dexterous poses, are also non-continuous. On the other hand, most other functions are continuous and continuously differentiable with respect to continuous variables, except for static and dynamic payload capacity assessments, which involve a minimum function and thus introduce non-differentiability when switching between limiting actuators. It is important to note that the mass evaluation function is the only one that provides an analytical gradient, as all other functions involve a degree of stochasticity. As a result, gradient-based algorithms will require the use of finite difference methods to approximate gradients.

Another key consideration is computational efficiency. For global performance evaluations, optimization-based functions are generally preferred over reachable workspace-based criteria. This is because the complete mapping of a criterion over the workspace is not interpretable by optimization algorithms, making it less suited for use as an objective function.

Lastly, the objective functions selected should be those that provide meaningful and practical benefits when fully optimized, rather than criteria that could more appropriately be treated as constraints. For instance, using reachability of a set of poses as an objective function may not be ideal. A better approach could be to pose the problem as optimizing another criteria while placing a constraint on achieving 100% reachability for the task.

In this thesis, optimization problems are conventionally posed as minimization problems. Thus, for maximization objectives, the negative of the function is minimized.

3.3.3 Constraint Selection

The selection of constraints follows similar guidelines to the selection of objective functions, with the additional step of specifying the maximum or minimum bounds that the criteria must not exceed. In this master's thesis, the negative-null form convention is used for constraints, meaning constraints are specified as less than or equal to zero. For instance, a function f constrained with an upper bound u, $f \leq u$ becomes : $g = f - u \leq 0$ and a function f with a lower bound l, $f \geq l$ becomes $l - f \leq 0$.

Many optimization algorithms, however, are not equipped to handle constraints directly. To address this, three different methods are implemented in the framework to convert the constrained problem into an unconstrained optimization problem. The unconstrained objective function $f'(\mathbf{x})$ is derived from the original objective function $f(\mathbf{x})$ and an inequality constraints vector $\mathbf{g}(\mathbf{x})$.

1. Death Penalty Method: This method penalizes infeasible solutions by assigning them a very large constant value k. The main advantage is its simplicity, but the downside is that the

algorithm will explore the infeasible space without any guidance on how to reduce constraint violations in highly constrained problems. The unconstrained objective function in this case becomes:

$$f'(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } \mathbf{g}(\mathbf{x}) < \mathbf{0} \\ k & \text{otherwise} \end{cases} \tag{3.12}$$

2. Weighted Penalty Method: This method adds a penalty term to the objective function, which is the weighted sum of constraint violations. While this approach allows the algorithm to explore the infeasible space more efficiently, it may lead to slightly infeasible solutions. The unconstrained objective function for this method is:

$$f'(\mathbf{x}) = f(\mathbf{x}) + w * \sum \max(0, g_i(\mathbf{x})) \tag{3.13}$$

where $g_i(\mathbf{x})$ is the value of the i-th constraint and w is the penalty weight.

3. Feasibility-First Method: In this method, the algorithm first prioritizes minimizing constraint violations to bring the solution into a feasible region before attempting to optimize the objective function. If any constraint is violated, the unconstrained function becomes the sum of all constraint violations, added to the maximum value of the objective function. The unconstrained objective function is then:

$$f'(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } \mathbf{g}(\mathbf{x}) < \mathbf{0} \\ f_{max} + \sum \max(0, g_i(\mathbf{x})) & \text{otherwise} \end{cases} \tag{3.14}$$

3.3.4 Algorithm Selection

The selection of algorithms to implement within the framework was determined using the survey of the manipulator optimization literature. The algorithms and number of occurrences in the literature are displayed in Table 3.4. Because, the majority of these algorithms are implemented in the pyGMO python toolbox [127], it was chosen as a reference framework within which Bayesian optimization and Complex optimization where additionally implemented. The following section provides a brief description of the best performing algorithms for the problems and performance comparison methodology presented in Chapter 4.

Table 3.4 Frequency of optimization algorithms in surveyed studies

Algorithm	**Count**
Genetic Algorithm	27
Complex	12
Sequential Quadratic Programming	10
Covariance Matrix Adaptation Evolution Strategy	6
Differential Evolution	4
Particle Swarm Optimization	4
Bayesian Optimization	2
Simulated Annealing	2
Ant Colony Optimization	1
Grey Wolf Optimization	1
Harris Hawk Optimization	1

Sequential Quadratic Programming

Sequential Quadratic Programming is the only gradient-based method implemented in the project. It is the most frequently used gradient-based method in manipulator optimization literature and regarded as one of the most efficient gradient-based methods for constrained optimization [128]. As its name indicates, the method consists in sequentially building quadratic problems (QP) using the Lagrangian of the constrained problem as well as an approximation of its Hessian. At each step, the solution to the QP problem provides a search direction and a line search strategy is used to execute a step with sufficient decrease in a merit function and update the Lagrange multipliers. This method is known to be very efficient with differentiable functions and well-conditioned problems. To reduce the likelihood of converging to local optima, it is often combined with a multi-start strategy, where the algorithm is initialized from multiple points in the design space. Figure 3.12 illustrates the first iterations of an SQP algorithm on a simple two-dimensional problem with inequality constraints.

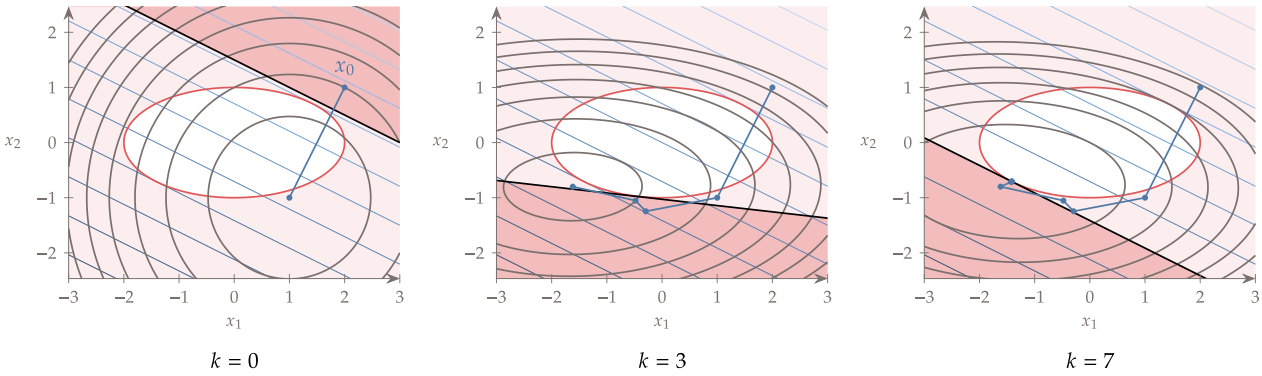


Figure 3.12 Iterations (k=0,3,7) of an SQP algorithm on a simple constrained problem, with the contours of the objective function in shades of blue, the constrained regions in red and the contours of the Lagrangian at each iteration in gray. [128]

Genetic Algorithm

A genetic algorithm is a population based-algorithm inspired by the principles of natural selection and biological evolution. It operates by evolving a population of candidate designs, typically represented as fixed-length chromosomes containing the design variables, over successive generations to optimize the objective function. The first population, set of N candidate designs, is initialized by randomly generating designs with variables within the specified bounds. The objective function, also referred to as fitness function, is evaluated for each of the designs in the population. Three main operators are then applied to the population to generate the next generation. The exact behaviour of the operators differs for different variations of genetic algorithms, the following is an overview of the implemented algorithm. First, a selection operator generates a population of N individuals with better performance. Tournament selection is used, generating N random subgroups of individuals and adding the best performing individual in each group to the population. Then, a crossover operator is applied, selecting N groups of two individuals from the population and combining their design variables with a chosen probability. Different methods can be used for the combination of design variables, in this project, simulated binary crossover is applied, producing an offspring by sampling values from a probability distribution centered around the parent designs. Finally, a mutation operator is applied to this population, randomly modifying the design variables of some of the designs in the population with a low probability. The result of these three steps is a new generation, on which the same operators can be applied again. As generations progress, the algorithm aims to converge towards a globally optimal design.

Covariance Matrix Adaptation Evolution Strategy

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [129] is also a population-based evolutionary algorithm. CMA-ES evolves a multivariate normal distribution over the search space, iteratively sampling candidate solutions from the distribution, selecting the best-performing individuals, and updating the distribution's parameters. The mean of the distribution is shifted towards better solutions through weighted recombination, while the covariance matrix is adapted to capture the shape of the objective function's landscape. Step-size control is managed separately using the evolution path technique, which adjusts the overall search scale based on the correlation of successive steps. CMA-ES is considered one of the most powerful black-box optimization algorithms due to its efficiency with small populations and its effectiveness in solving non-linear, non-convex, and ill-conditioned problems. Figure 3.13 illustrates the evolution of the distribution using CMA-ES on a convex 2-dimensional problem.

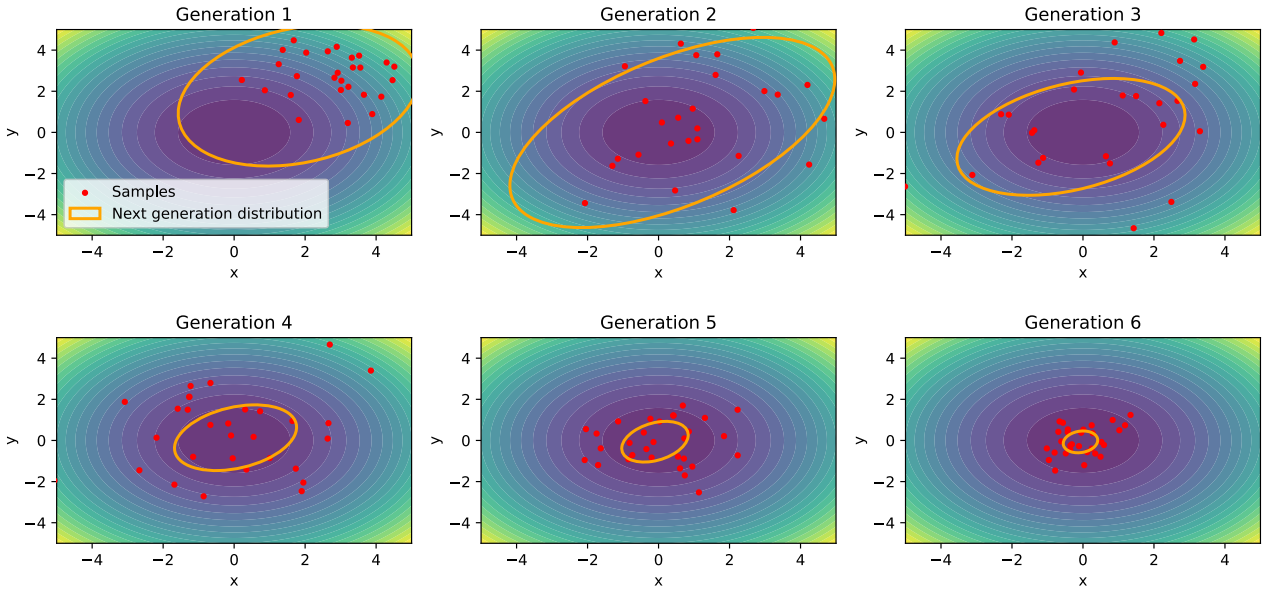


Figure 3.13 Evolution of the CMA-ES optimization process over six generations on a 2D convex function. Contours show the objective landscape, red dots represent sampled candidates, and orange ellipses indicate the updated search distribution.

Bayesian Optimization

Bayesian optimization is a global optimization method designed to minimize the number of function evaluations needed to find an optimum. It operates by constructing a surrogate probabilistic model of the objective function which estimates the value of the function for

non-evaluated designs and quantifies the uncertainty of the prediction. At each step, an acquisition function is optimized to determine the next design to evaluate, balancing between exploration, evaluating designs with high uncertainty, and exploitation, evaluating designs where the surrogate model predicts improvement. The surrogate model is then updated with the information on the newly evaluated design and the cycle repeats. This method is particularly efficient for computationally expensive black-box functions. It was also extended to multi-objective optimization by Ozaki et al. [130]. Although rarely used in robotic optimization literature, it is implemented to compare with other well-known methods. An example of Bayesian optimization on a 1-D multi-modal function is illustrated in Figure 3.14, it illustrates the balance between exploration and exploitation, displaying the surrogate model and its confidence interval.

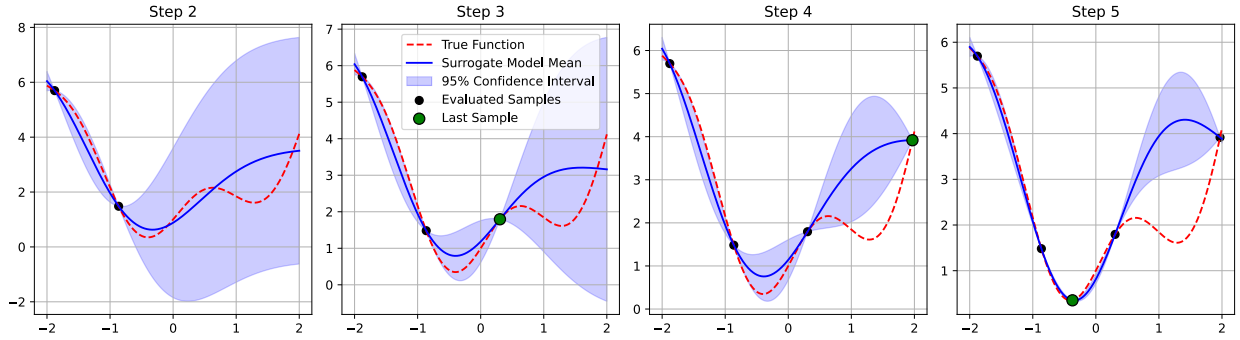


Figure 3.14 Illustration of four iterations of Bayesian optimization on a one-dimensional multi-modal function, with the surrogate model mean and confidence interval in blue, the real function in red and the sampled points in black.

Non-Dominated Sorted Genetic Algorithm II

The most used multi-objective optimization algorithm in the literature is NSGA-II. It extends the classical genetic algorithm by considering multiple fitness functions.

The process begins identically to the GA with the initialization of a population of N candidate solutions, randomly generated within the predefined variable bounds. Each design is evaluated using the objective function and the three variations of the GA genetic operators are applied.

In the selection phase, candidate designs are ranked using non-dominated sorting to form a series of fronts (as described in Chapter 2), where each front i is dominated only by the preceding front *i-1*. Within each front, individuals are further ranked based on their crowding distance to promote diversity. The top $\frac{N}{2}$ individuals are selected to form the basis of the

next generation.

Next, the crossover operator creates $\frac{N}{2}$ offspring by pairing individuals and recombining their design variables with a given probability.

Finally, the mutation operator is applied identically to the GA procedure, ensuring genetic diversity in the population. The resulting new generation is then subjected to the same sequence of operations in the subsequent iteration. Over time, the algorithm converges towards the true Pareto front, ideally yielding a diverse set of high-quality, non-dominated solutions. The evolution of the population with NSGA-II over 200 generations on an example multi-objective problem is illustrated in Figure 3.15.

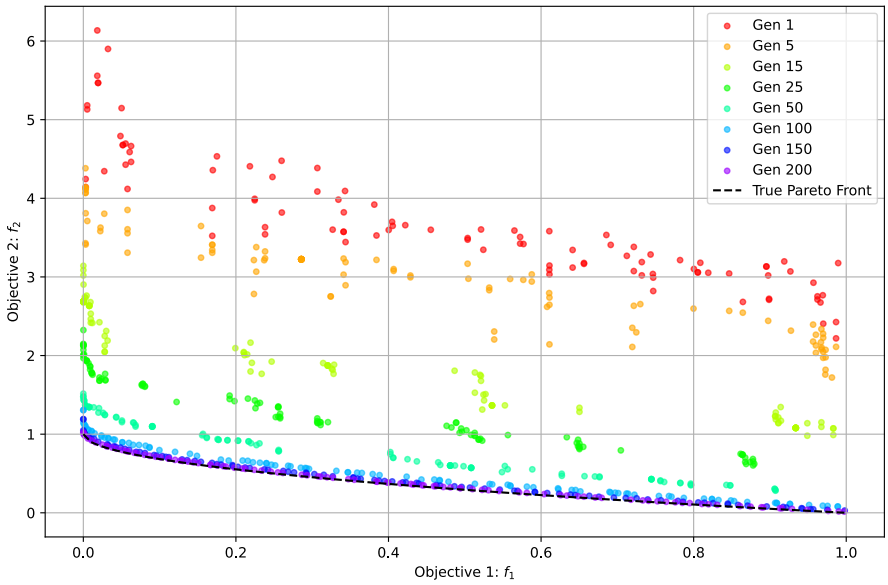


Figure 3.15 Evolution of the population of the NSGA-II algorithm on an example multi-objective problem.

To summarize this chapter, Sections 3.1 and 3.2 address **SO-1**, and together with Section 3.3, they develop the complete methodology to achieve **SO-2**. The next chapter will use the results obtained from the framework to address **SO-3**, by constructing a set of representative problems and identifying the most efficient algorithms.

CHAPTER 4 RESULTS

The methodology outlined in the previous chapter enabled the development of a general and versatile framework designed to assist engineers in the conceptual design of serial manipulators, incorporating both direct and inverse design strategies. This chapter focuses on demonstrating the application of the proposed framework across a range of inverse design problems. The objective is twofold: to highlight the flexibility of the framework and to assess the performance of various optimization algorithms when applied to different classes of problems.

The structure of the chapter is as follows. First, in Section 4.1, the direct design approach is used to evaluate the performance of a reference manipulator design using the performance metrics introduced in Section 3.2. In particular, task-specific criteria are assessed across a set of defined tasks, which subsequently serve as benchmarks for the inverse design results. Next, in Section 4.2, a classification scheme for optimization problems is introduced, followed by the detailed presentation of six representative problem cases. Finally, Section 4.3 discusses the results obtained from optimizing each problem using a range of optimization algorithms, providing comparative insights into their effectiveness.

4.1 Direct Design Performance Analysis of the Reference Design

Before attempting to construct and optimize inverse design problems, the performance of a reference design is evaluated. In this aim, a model of the Kinova GEN-3 collaborative robot is developed. Because the values of the actuator specifications are not completely publicly available, commercially available motor and reducers were selected to obtain similar torque and speed performance. Similarly, the structural parameters to construct the cylindrical shell models were estimated to obtain mass properties similar to the available information. The complete parametrized model is illustrated in Annex A. The following section analyzes the global performance of the robot as well as its performance on three tasks, illustrating the different task types presented in Section 3.2.1.

1. Task 1 - Position-based : A set of 16 discrete Cartesian positions filling a rectangular area of 50cm $\times$ 70cm in the x–z plane. The rectangle is centered at x = 45cm and z = 35cm. This task could be used to assess the performance of a design with a workspace presenting axial symetry.

2. Task 2 - Path-based : A pick and place task defined by three poses with a downwards

pointing orientation, defined by a rotation of 180° around the X-axis of the reference frame:

(a) A picking pose : x = 50cm, y = 0cm and z = 0cm

(b) A transition pose : x = 25cm, y = 25cm and z = 20cm

(c) A placing pose : x = 0cm, y = 50cm and z = 0cm

This basic representation of a typical robot task is used to evaluate the manipulator's speed capacity.

3. Task 3 - Trajectory-based : This task builds upon Task 2 by incorporating time intervals at each pose to define a time-parametrized trajectory:

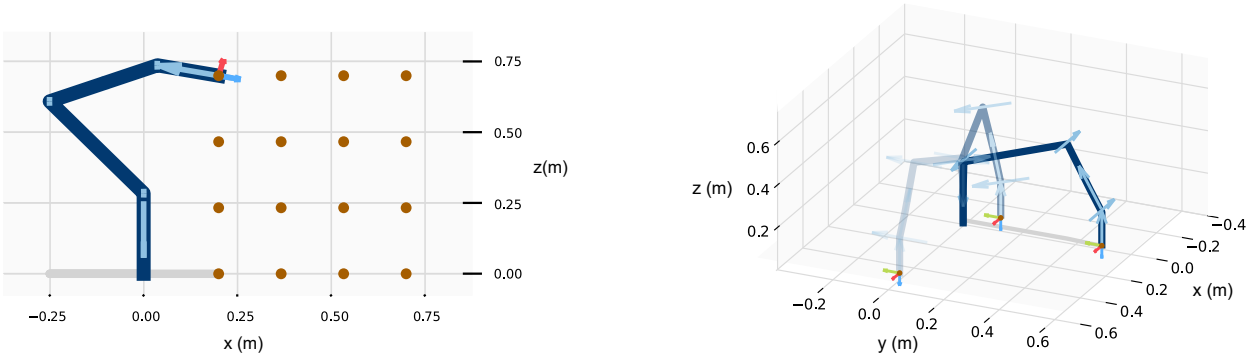
(a) t = 0s: Picking pose.

(b) t = 0.4s: Transition pose.

(c) t = 0.8s: Placing pose.

This task is used as an example to evaluate the robot's dynamic performance.

The Cartesian points and poses defining each of the three tasks are illustrated in Figure 4.1. For each task, all relevant performance criteria are evaluated, and the corresponding results, including computation times, are summarized in Table 4.1. All computations were performed on a laptop equipped with an Intel® Core™ i7-12700H processor.



(a) Task 1 points

(b) Task 2/3 poses

Figure 4.1 Illustration of the Cartesian points and poses defining the three reference tasks used for performance evaluation.

Table 4.1 Results and computation times for the performance evaluation of the reference Kinova GEN-3 manipulator model using the direct design approach on global and task-specific performance criteria.

	Performance Criteria	**Result**	**Computation Time**
Global			
	Mass	8.7 kg	< 1 ms
	Maximum Reach[1]	0.892 m	50 ms
	Reachable Workspace Volume[2]	2.9 m^3	10 s
	SLI[2]	0.83	10 s
	GCI[2]	0.063	80 s
	Max. Static Static Payload Capacity[1]	1.14 kg	2 s
	Min. Natural Frequency[1 5]	8.91 Hz	13 s
	Max. End Effector Deflection[1 5]	4.05 mm	4 s
Task 1			
	Reachability	100%	50 ms
	Minimum Manipulability[4]	0.008	11 s
	Average Dexterity[3]	62%	9 s
	Min. Natural Frequency[5]	8.57 Hz	34 s
	Max. Static Payload Capacity	1.9 kg	13 s
	Max. End Effector Deflection[5]	3.45 mm	31 s
Task 2			
	Reachability[*]	100%	10 ms
	Minimum Manipulability[*4]	0.105	50 ms
	Min. Natural Frequency[*5]	9.56 Hz	2 s
	Max. Static Payload Capacity[*]	3.43 kg	300 ms
	Max. End Effector Deflection[5]	2.38 mm	600 ms
	Min. Cycle Time	0.78 s	2 s
	Max. End Effector Velocity	1.38 m/s	2 s
Task 3			
	Max. End Effector Deflection[5]	4.3 mm	2 s
	Trajectory Feasibility	Feasible	1 s
	Peak Power Consumption	91.76 W	1 s
	Mechanical Work	40.45 J	1 s
	Max. Dynamic Payload Capacity	3.45 kg	2 s

[1] multi-start with 10 starts.

[2] with 10^7 joint space samples.

[3] with 30 tested orientations.

[4] with inverse conditioning number index.

[5] with task maximum static payload.

[*] identical for task 2 and 3.

The obtained results will be used as a reference in Section 4.3, where the performance of the optimized designs resulting from the inverse design problems will be compared to this reference. Additionally, the computation times presented in Table 4.1 demonstrate the efficiency of the direct design framework, ranging from less than one millisecond to approximately 80 seconds.

4.2 Construction of Representative Inverse Design Problems

Building on the performance of the reference design, the inverse design approach is used to optimize the robot's design based on selected performance criteria. This section aims to define a set of representative inverse design problems, which will be used to evaluate a range of optimization algorithms. The goal is to address sub-objective **SO-3** by identifying suitable algorithms to each problem type.

To support generalization, a classification scheme is developed based on the formulation of inverse design problems. The proposed scheme draws inspiration from the framework introduced by Martins and Ning [131], which considers three key formulation characteristics: the type of design variables (continuous, discrete, or mixed), the number of objective functions (single- or multi-objective), and the presence of constraints (constrained or unconstrained problems). For simplicity, problems with discrete or mixed design variables are grouped into a single category, thereby distinguishing between problems with only continuous variables and those that include at least one discrete variable.

In addition to these three formulation characteristics, the dimensionality of the design space is introduced as a fourth criterion, given that some algorithms are known to scale poorly with higher dimensions. Following a common rule of thumb [131], problems with fewer than ten design variables are classified as low-dimensional, while those with ten or more are considered high-dimensional. The complete classification scheme, incorporating all four criteria, is illustrated in Figure 4.2.

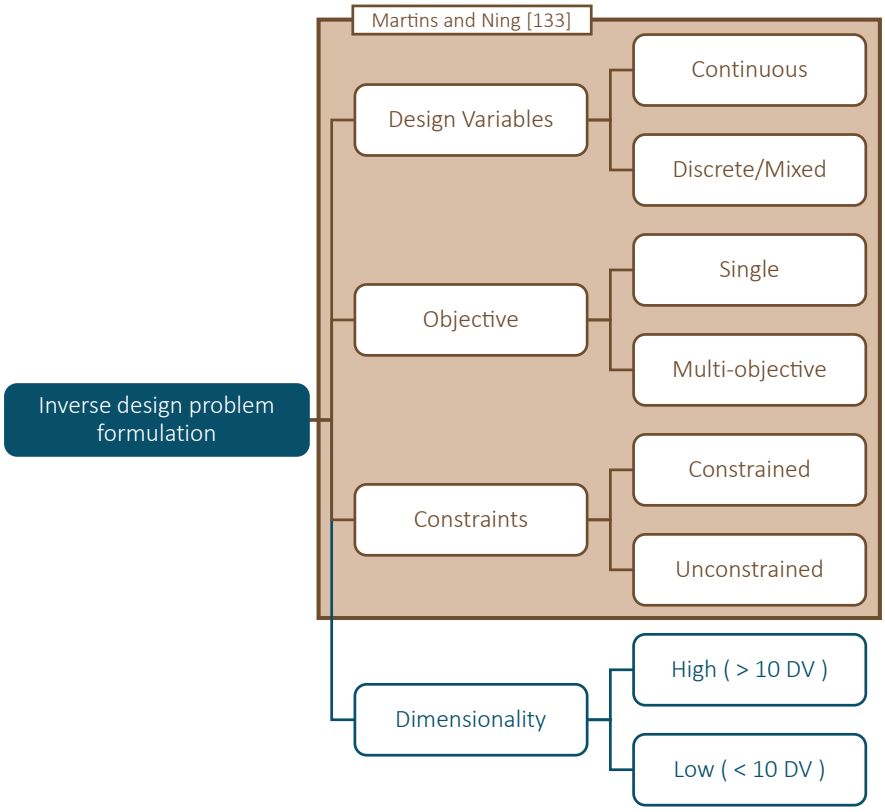


Figure 4.2 Classification of inverse design problems based on design variables, objective function(s), constraints, and dimensionality. The framed characteristics are adapted from Martins and Ning [131], while dimensionality is added to reflect problem complexity.

Based on this classification, a set of inverse design problem classes is defined. By combining the variable type, constraint presence, and dimensionality criteria, $2^3 = 8$ distinct classes are obtained. For simplification, two of these classes, those involving unconstrained problems with only continuous variables, are excluded from the study, as no relevant or non-trivial problems could be formulated to represent them.

For each of the six remaining classes, a representative problem is constructed to evaluate algorithm performance across problem formulations. In addition, four multi-objective variants are derived from the constrained problems by converting one of the constraints into a second objective function. The resulting set of problem classes is illustrated in Figure 4.3.

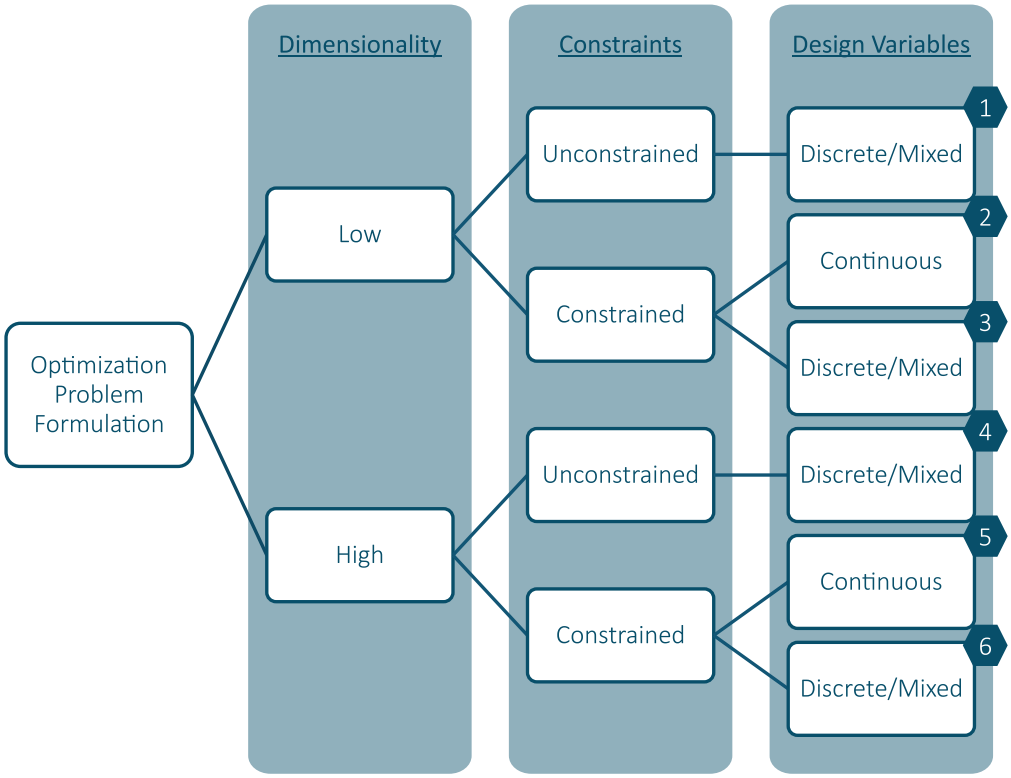


Figure 4.3 Inverse design problem formulation classes, numbered from one to six, derived from the combination of characteristics from the introduced classification.

4.2.1 Problem 1 (Class 1) : Actuator Selection for Payload Maximization

The first class encompasses unconstrained optimization problems involving fewer than ten variables, including discrete design variables. To illustrate this class, the following representative problem is presented. The robot is intended for operation on an assembly line, where it performs a pick-and-place task (Task 3). The objective of the inverse design problem is to determine the optimal selection of actuators to be assigned to each joint of the robot, in order to maximize its dynamic payload capacity (see Section 3.2.1) along the specified trajectory. The available actuators are drawn from the catalog provided in Annex B. The formal problem formulation is as follows:

$$\begin{aligned} &\underset{\mathbf{y}}{\text{maximize}} && \mathcal{P}_{\text{dyn}}(\mathbf{y}) \\ &\text{where} && y_i \in \{\texttt{Small}, \texttt{Medium}, \texttt{Large}, \texttt{XLarge}\} \quad \forall i \in \{1, 2, \ldots, 6\}, \end{aligned} \tag{4.1}$$

Where $\mathcal{P}_{\text{dyn}}(\mathbf{y})$ is the dynamic payload capacity over the trajectory for a robot with a set of actuators $\mathbf{y}$.

4.2.2 Problem 2 (Class 2) : Link Length Optimization for Global Performance

The second class includes constrained optimization problems involving fewer than ten continuous variables. The representative problem for this class focuses on optimizing the global performance of the robot. Specifically, the objective is to determine the optimal link lengths that maximize the robot's static payload capacity (see Section 3.2.1) throughout its entire workspace.

To ensure a sufficiently large workspace and adequate manipulability, constraints are imposed on the robot's maximum reach (see Section 3.2.2) and GCI (see Section 2.2.1), which must exceed 1 meter and 0.063 (reference design GCI), respectively. Each link's length is defined as the largest translational component within its ETS representation. Since the lengths of the first two links do not influence the considered performance criteria, they are excluded from the optimization.

The formal problem formulation is as follows:

$$
\begin{aligned}
\underset{\mathbf{x}}{\text{maximize}} \quad & \mathcal{P}_{static}(\mathbf{x}) \\
\text{subject to} \quad & \mathcal{R}(\mathbf{x}) \geq 1\text{m} \\
& \text{GCI}(\mathbf{x}) \geq 0.063 \\
\text{where} \quad & \mathbf{x} = \{l_3, l_4, l_5, l_6, l_7\} \\
& l_3 = -T_{y,3} \in [0.05\text{m}, 1\text{m}], \\
& l_4 = \quad T_{y,4} \in [0.05\text{m}, 1\text{m}], \\
& l_5 = -T_{z,5} \in [0.05\text{m}, 1\text{m}], \\
& l_6 = \quad T_{y,6} \in [0.05\text{m}, 1\text{m}], \\
& l_7 = -T_{z,7} \in [0.05\text{m}, 1\text{m}],
\end{aligned}
\tag{4.2}
$$

Where $\mathcal{P}_{static}(\mathbf{x})$ is the static payload capacity of the robot over its workspace, $\mathcal{R}(\mathbf{x})$ is its maximum reach, $\text{GCI}(\mathbf{x})$ is its Global Conditioning Index for a set of link lengths $\mathbf{x}$ and $T_{y,i}$ and $T_{z,i}$ are the y and z translational components of the i-th link.

This problem can be converted to a multi-objective problem by converting maximum reach to a secondary objective. This formulation will allow the visualization of the trade-off between maximum reach and payload capacity.

4.2.3 Problem 3 (Class 3) : Energy-Efficient Actuator Selection under Payload Constraint

The third class comprises constrained optimization problems with fewer than ten variables, including discrete design variables. The representative problem for this class builds on the same pick-and-place trajectory (Task 3) introduced in Problem 1. However, the objective here is to identify the most energy-efficient combination of actuators capable of executing the task with a specified payload.

The goal is to minimize the total mechanical work (see Section 3.2.1) generated by the actuators during the trajectory. A constraint is imposed to ensure that the robot can support a 7 kg payload throughout the execution of the task.

The formal problem formulation is presented as follows:

$$\begin{aligned} \underset{\mathbf{y}}{\text{minimize}} \quad & \mathcal{W}(\mathbf{y}) \\ \text{subject to} \quad & \mathcal{P}_{\text{dyn}}(\mathbf{y}) \geq 7\text{kg} \\ \text{where} \quad & y_i \in \{\texttt{Small}, \texttt{Medium}, \texttt{Large}, \texttt{XLarge}\} \quad \forall i \in \{1, 2, \ldots, 6\}, \end{aligned} \tag{4.3}$$

Where $\mathcal{W}(\mathbf{y})$ is the mechanical work produced by the robot over the trajectory for a set of actuators $\mathbf{y}$.

This problem can be converted to a multi-objective problem by converting dynamic payload capacity to a secondary objective allowing the understanding of the tradeoffs between energy efficiency and payload capacity.

4.2.4 Problem 4 (Class 4) : Motor and Reducer Selection for Cycle Time Minimization

The fourth class encompasses unconstrained optimization problems with more than ten variables, including discrete design variables. The representative problem for this class seeks to identify the optimal combination of motors and reducers for each joint of the robot in order to minimize the minimum achievable cycle time (see Section 3.2.1) for completing the pick-and-place path (Task 2).

While formally categorized as unconstrained, the problem includes an implicit feasibility condition: any design that cannot compensate gravity-induced torques is excluded from consideration. The selection of motors and reducers is made from a catalog comprising 27 reducers and 8 motors (see Annex C). The problem formulation is described as follows:

$$
\begin{aligned}
&\underset{\mathbf{y}}{\text{minimize}} \quad \mathcal{C}_{\mathrm{t}}(\mathbf{y}) \\
&\text{where} \qquad y_i \in \begin{cases} \{1, \ldots, \ 8\} & \forall i \in \{1, 2, \ldots, 6\}, \\ \{1, \ldots, 27\} & \forall i \in \{7, 8, \ldots, 12\}, \end{cases}
\end{aligned} \tag{4.4}
$$

Where $\mathcal{C}_{\mathrm{t}}(\mathbf{y})$ is the minimum cycle time over the trajectory for a robot with a set of motors and reducers $\mathbf{y}$.

4.2.5 Problem 5 (Class 5) : Lightweight Dexterous Robot Design via Complete Kinematic Optimization

The fifth class represents constrained optimization problems with more than ten continuous variables. The problem selected to represent this class focuses on designing a lightweight and highly dexterous robot for a position-based task (Task 1). The design objective is to minimize the robot's mass while ensuring full dexterity (see Section 2.2.1) at each point of the task.

To achieve this, the full set of the robot's ETS parameters is optimized. Analogous to Denavit-Hartenberg (DH) parameters, the optimization is limited to the translational components T_y, T_z and R_x components of the ETS. In order to simplify the dexterity analysis, the T_y and R_x components of the last link are omitted. Similarly, the T_y and R_x components of the first link components of the first link are fixed to preserve axial symmetry and avoid radial translation of the robot's base. The problem formulation is described as follows:

$$
\begin{aligned}
&\underset{\mathbf{x}}{\text{minimize}} \quad \mathcal{M}(\mathbf{x}) \\
&\text{subject to} \quad \mathcal{D}(\mathbf{x}) = 1 \\
&\text{where} \qquad \mathbf{x} = \{T_{y,1}, T_{y,2}, T_{z,2}, R_{x,2}, \ldots, T_{y,6}, T_{z,6}, R_{x,6}, T_{z,7}\} \\
&\qquad\qquad T_{y,i} \in [-1\mathrm{m}, 1\mathrm{m}] \quad \forall i \in \{2, \ldots, 6\}, \\
&\qquad\qquad T_{z,i} \in [-1\mathrm{m}, 1\mathrm{m}] \quad \forall i \in \{1, \ldots, 7\}, \\
&\qquad\qquad R_{x,i} \in [-180^\circ, 180^\circ] \quad \forall i \in \{2, \ldots, 6\},
\end{aligned} \tag{4.5}
$$

Where $\mathcal{M}(\mathbf{x})$ is the mass of the robot, and $\mathcal{D}(\mathbf{x})$ is the average dexterity index across all task points for the design defined by the set of ETS parameters $\mathbf{x}$.

This problem can be reformulated as a multi-objective optimization problem by treating the average dexterity index as a secondary objective. This approach enables a clearer visualization of the trade-off between the robot's lightweight design and its dexterity.

4.2.6 Problem 6 (Class 6) : Comprehensive High-Performance Manipulator

The sixth and final class represents constrained inverse design problems involving more than ten design variables, including both continuous and discrete variables. This class accommodates the full range of problem characteristics, enabling the formulation of a comprehensive inverse design problem that simultaneously considers global and task-specific performance.

The objective is to design a lightweight robot with good global manipulability, capable of reaching Task 1 and executing Task 3 with a specified payload. Additionally, it should be capable of supporting a specific payload across its entire workspace. The optimization goal is to minimize the robot's total mass, subject to several performance constraints: full reachability (100%, see Section 2.2.1) of both Task 1 and Task 3, a global static payload capacity (see Section 3.2.1) greater then 1 kg, a dynamic payload capacity (see Section 3.2.1) of at least 8 kg for Task 3, and a GCI (see Section 2.2.1) exceeding 0.07.

To solve this problem, both the link lengths and the selection of motors and reducers at each joint are optimized. The problem is formally defined as:

$$
\begin{aligned}
\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \quad & \mathcal{M}(\mathbf{x},\mathbf{y}) \\
\text{subject to} \quad & \mathcal{R}_{1,3}(\mathbf{x}) = 100\% \\
& \mathcal{P}_{static}(\mathbf{x},\mathbf{y}) \geq 1\text{kg} \\
& \mathcal{P}_{dyn}(\mathbf{x},\mathbf{y}) \geq 7\text{kg} \\
& \text{GCI}(\mathbf{x}) \geq 0.07 \\
\text{where} \quad & \mathbf{x} = \{l_1, l_2, l_3, l_4, l_5, l_6, l_7\} \\
& l_1 = \quad T_{z,1} \in [0.05\text{m}, 1\text{m}], \\
& l_2 = -T_{z,2} \in [0.05\text{m}, 1\text{m}], \\
& l_3 = -T_{y,3} \in [0.05\text{m}, 1\text{m}], \\
& l_4 = \quad T_{y,4} \in [0.05\text{m}, 1\text{m}], \\
& l_5 = -T_{z,5} \in [0.05\text{m}, 1\text{m}], \\
& l_6 = \quad T_{y,6} \in [0.05\text{m}, 1\text{m}], \\
& l_7 = -T_{z,7} \in [0.05\text{m}, 1\text{m}], \\
& y_i \in \begin{cases} \{1, \ldots, \ \ 8\} & \forall i \in \{1, 2, \ldots, 6\}, \\ \{1, \ldots, 27\} & \forall i \in \{7, 8, \ldots, 12\}, \end{cases}
\end{aligned}
\tag{4.6}
$$

Where $\mathcal{M}(\mathbf{x},\mathbf{y})$ denotes the robot's mass, $\mathcal{R}_{1,3}(\mathbf{x})$ is the reachability of tasks 1 and 3,

$\mathcal{P}_{\text{static}}(\mathbf{x})$ is the global static payload capacity, $\mathcal{P}_{\text{dyn}}(\mathbf{x})$ is the dynamic payload capacity on task 3 and GCI($\mathbf{x}$) is the Global Conditioning index of a design defined by a set of link lengths $\mathbf{x}$ and motor and reducer selections $\mathbf{y}$.

Similarly to the previous constrained problems, this problem can be reformulated into a multi-objective problem by converting the dynamic payload capacity of the robot on task 3 to a secondary objective. The remaining constraints will maintain the problem constrained in order to only analyze designs with good global payload capacity and GCI.

4.3 Algorithm Performance Comparison

To identify the most suitable algorithm for optimizing each class of inverse design problem, a range of optimization algorithms is applied to each problem instance.

For mono-objective problems, the algorithms evaluated include: GA, DE, CMA-ES, PSO, Bayesian optimization, Complex algorithm, ACO, and SQP. It is important to note that SQP is applied only to Problems 2 and 4, as it requires continuous design variables. Algorithm performance is assessed based on two criteria: (1) the objective value of the best feasible solution found, and (2) the number of function evaluations required to reach that solution.

For multi-objective problems, the algorithms tested include: NSGA-II, Multi-Objective Ant Colony Optimization (MACO), Multi-Objective Evolutionary Algorithm with Decomposition (MOEAD), Non-Dominated Sorting Particle Swarm Optimization (NSPSO) and multi-objective Bayesian optimization. Performance is measured using the hypervolume indicator, which quantifies the size of the dominated region in the objective space bounded by a reference point. In the case of bi-objective problems, this corresponds to the union of areas defined by rectangles extending from each Pareto-optimal point to the reference point. This metric is particularly relevant as it captures both the convergence toward the true Pareto front and the diversity of the solution set [132]. The chosen reference point is defined by the worst objective values obtained across all solutions.

Since not all objectives in the multi-objective problems are formulated as minimization objectives, the resulting Pareto fronts in the following sections may not conform to the conventional bottom-left dominant shape. This visualization was chosen to preserve the interpretability of the objective values, rather than applying a negation to the maximized objectives.

Since SQP is the only algorithm in this study that natively handles constraints, all constrained problems are reformulated to a unconstrained formulation using the weighted penalty method presented in Section 3.3.3 for the remaining algorithms.

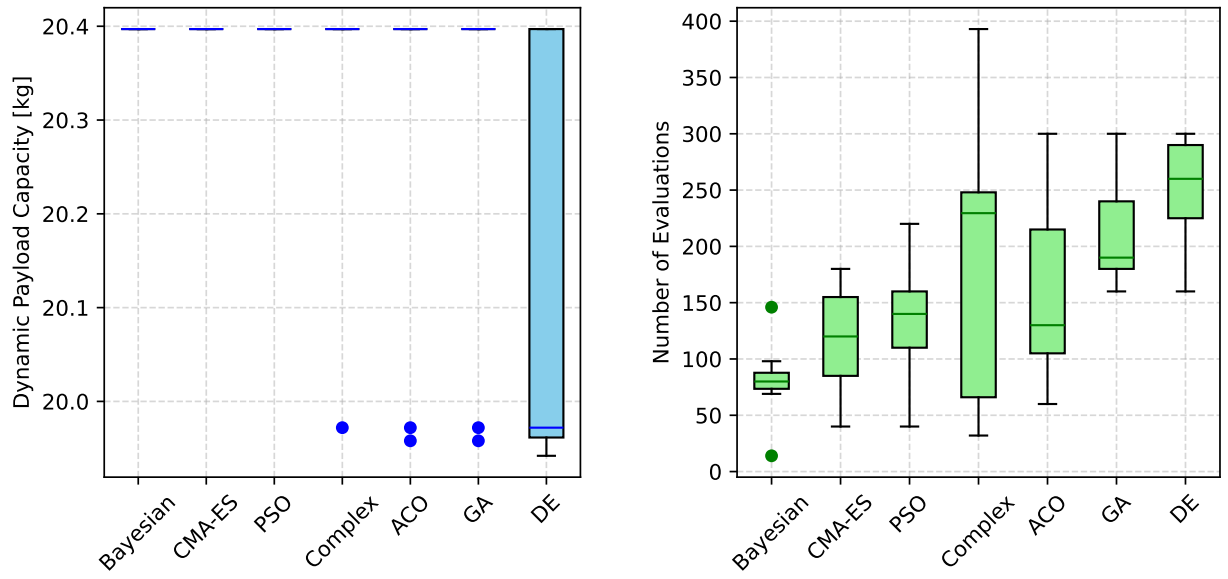
To ensure a fair comparison, all algorithms are subject to the same maximum number of function evaluations as a termination criterion. Additionally, population-based algorithms are initialized with the same population size and number of generations. Each algorithm is executed ten times per problem using different random seeds to account for stochastic variability. The results for each problem are detailed in the following sections.

4.3.1 Problem 1

As presented in Section 4.2.1, Problem 1 aims to identify the optimal set of actuators to maximize the payload capacity of a robot over a trajectory defined by Task 3. Figure 4.4 shows box plots illustrating the distribution of the best objective values and the number of function evaluations for each of the seven algorithms across ten independent runs.

Among the evaluated algorithms, the Bayesian optimization method, CMA-ES, and PSO demonstrated the most consistent performance, with all independent runs converging to the same optimized solution (Table D.1). As a result, the box plots for these algorithms appear as horizontal lines, since there is no variation in their outcomes across runs. In terms of efficiency, the Bayesian algorithm was the most effective, reaching the best design with an average of only 81 function evaluations. The payload capacity of 20.4 kg of the optimized design represents a 590% improvement over the reference design.

Although the problem's formulation and low dimensionality may suggest simplicity, an exhaustive search would have required 4096 evaluations. This comparison highlights the substantial computational savings achieved through the use of these optimization algorithms.



(a) (b)

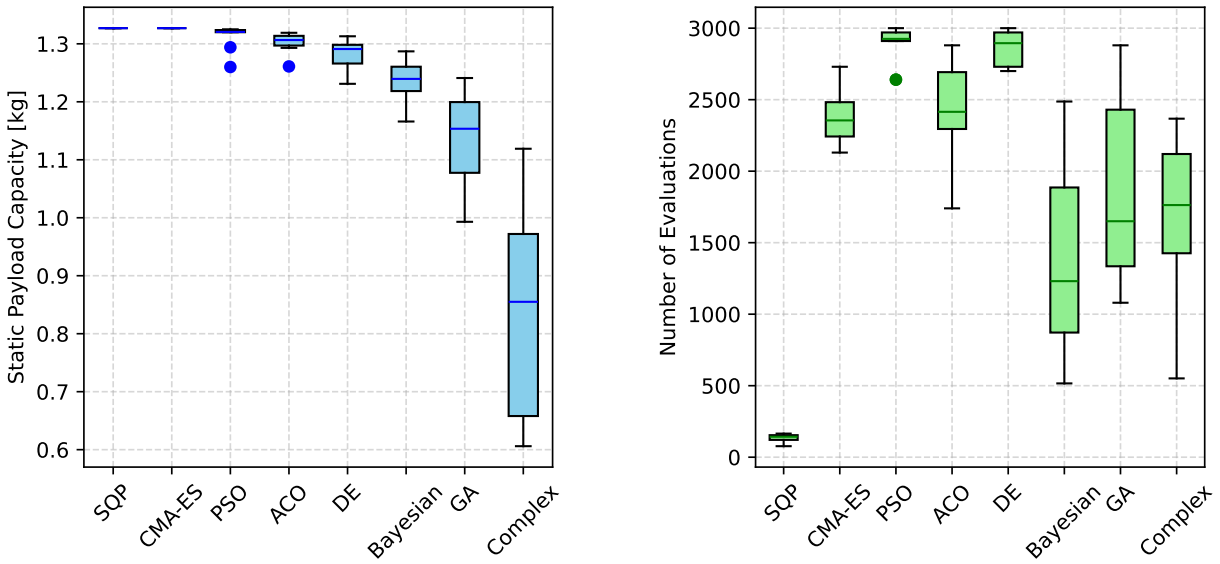
Figure 4.4 Box plots showing the performance of each algorithm over 10 independent runs on Problem 1. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

4.3.2 Problem 2

As presented in Section 4.2.2, Problem 2 is formulated to optimize link lengths of the robot to maximize payload capacity across its entire workspace, subject to constraints on both maximum reach and global manipulability. Figure 4.5 shows box plots illustrating the distribution of the best objective values and the number of function evaluations required to reach the best design for each of the eight algorithms over ten independent runs.

The highest feasible objective value was obtained by the SQP and CMA-ES algorithms, both converging to a static payload capacity of 1.33 kg, the corresponding design is detailed in Table D.2. Compared to the reference configuration, this design improves both the maximum reach and payload capacity while maintaining the same GCI value.

In terms of algorithmic performance, SQP clearly outperforms the others, converging towards the best design with approximately 18 times fewer function evaluations than CMA-ES. Although this comparison slightly underestimates the true number of evaluations used by SQP, as line search evaluations are not counted, it nonetheless demonstrates the algorithm's superior efficiency.



(a) (b)

Figure 4.5 Box plots showing the performance of each algorithm over 10 independent runs on Problem 2. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

The multi-objective formulation introduced in Section 4.2.2 enables the analysis of trade-offs

between maximum reach and payload capacity. Figure 4.6 presents the distribution of Pareto front hypervolumes across ten independent runs for each algorithm, as well as the best Pareto front obtained by each method, based on the hypervolume criterion.

Among the evaluated algorithms, the Bayesian optimization method and MOEAD yield the best performance. Both converge toward similar Pareto fronts characterized by well-distributed, non-dominated solutions. In contrast, NSPSO exhibits the weakest performance, producing designs that clustered within a narrow region and are entirely dominated by the Pareto fronts of the other algorithms.

From a user perspective, the most relevant portion of the Pareto front lies in the region with strictly positive payload capacities. In this region, the Pareto fronts obtained by the best-performing algorithms reveal a clear trade-off between a maximum payload capacity of 2.55 kg and a maximum reach of 1.46 m. As a reminder, a negative global static payload capacity means that the robot cannot resist gravitational forces in a part of its workspace, the negative values facilitate the exploration of the design space by optimization algorithms (see Section 3.2.1).

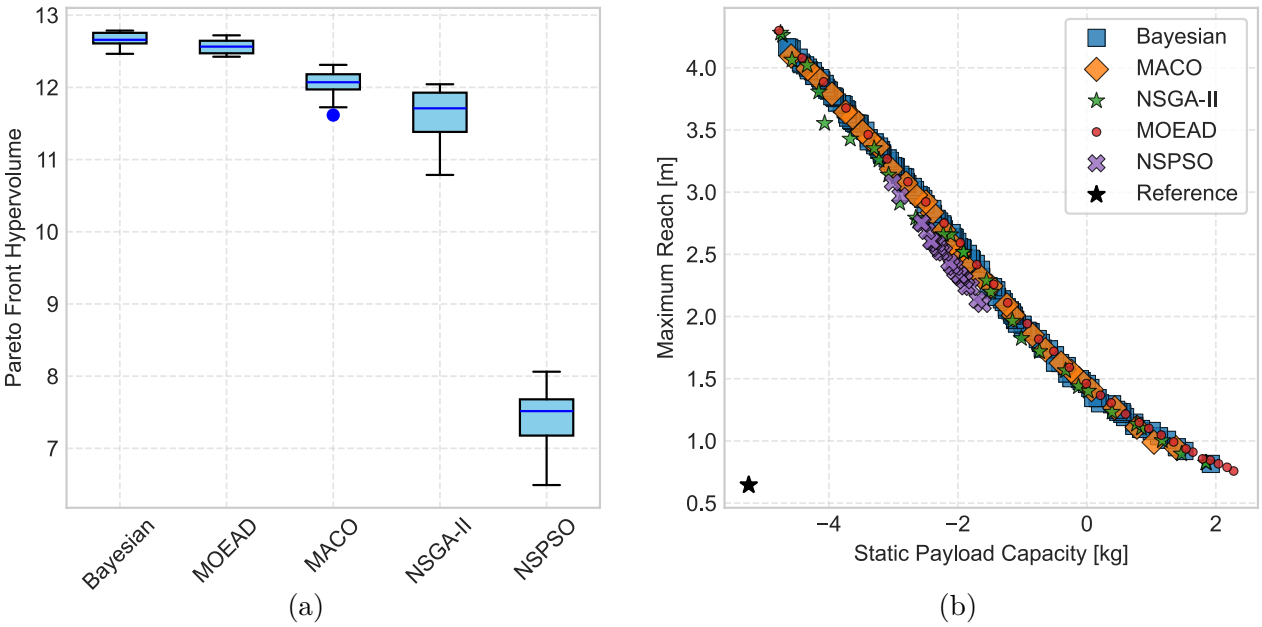


Figure 4.6 Performance of each algorithm on the multi-objective formulation of Problem 2. (a) Distribution of Pareto front hypervolumes across 10 independent runs. (b) Best Pareto front achieved by each algorithm, selected based on the hypervolume criterion.

4.3.3 Problem 3

As introduced in Section 4.2.3, Problem 3 seeks to identify the optimal actuators to minimize the mechanical work required for executing Task 3, subject to a constraint on payload capacity. Figure 4.7 presents box plots showing the distribution of the best objective values and the number of function evaluations required to reach them, across ten independent runs of each of the seven evaluated algorithms.

The best design, reported in Table D.3, achieves a mechanical work of 37.53 J while maintaining a dynamic payload capacity of 8.09 kg. Compared to the reference design, this configuration reduces energy consumption during Task 3 by 7% while supporting more than twice the payload.

As with Problem 1, all algorithms are able to identify the best design in at least one of the independent runs. However, Bayesian optimization, CMA-ES, ACO, and GA consistently reach the best design in all runs. In terms of efficiency, Bayesian optimization is again the most effective, requiring the fewest function evaluations on average, followed closely by CMA-ES and ACO.

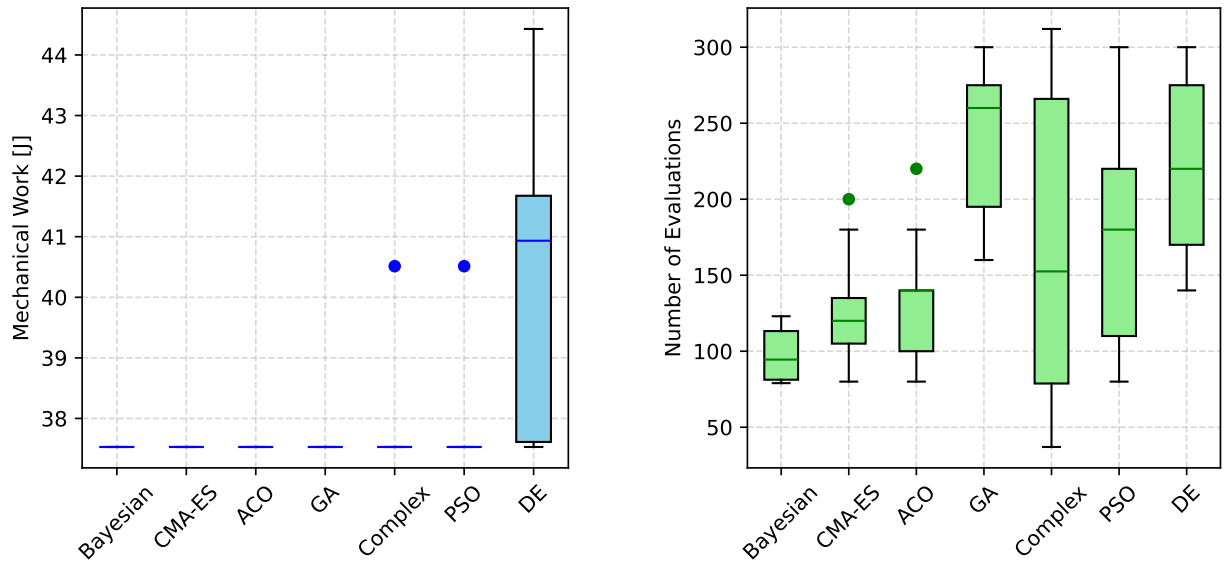


Figure 4.7 Box plots showing the performance of each algorithm over 10 independent runs on Problem 3. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

The multi-objective formulation introduced in Section 4.2.3 facilitates the exploration of

trade-offs between energy consumption (mechanical work) and payload capacity. Figure 4.8 presents the distribution of Pareto front hypervolumes across ten independent runs for each algorithm, along with the best Pareto front obtained by each method, selected based on the hypervolume criterion.

Among the evaluated algorithms, the Bayesian optimization method demonstrates the best performance, consistently producing Pareto fronts with the highest hypervolumes and the greatest number of non-dominated solutions. Due to the low dimensionality of the problem, the total number of possible non-dominated designs is limited, making it increasingly important for an algorithm to recover all of them. While most algorithms, except NSPSO, identify some solutions on the true Pareto front, Bayesian optimization retrieves the largest proportion.

This analysis also provides insight into why the payload capacity constraint in the mono-objective formulation is not active in the optimized design. The Pareto front reveals that the next non-dominated design with lower mechanical work corresponds to a payload capacity of 6.25 kg, which violates the 7 kg constraint. Therefore, the mono-objective result reflects a trade-off where the objective cannot be lowered without compromising feasibility.

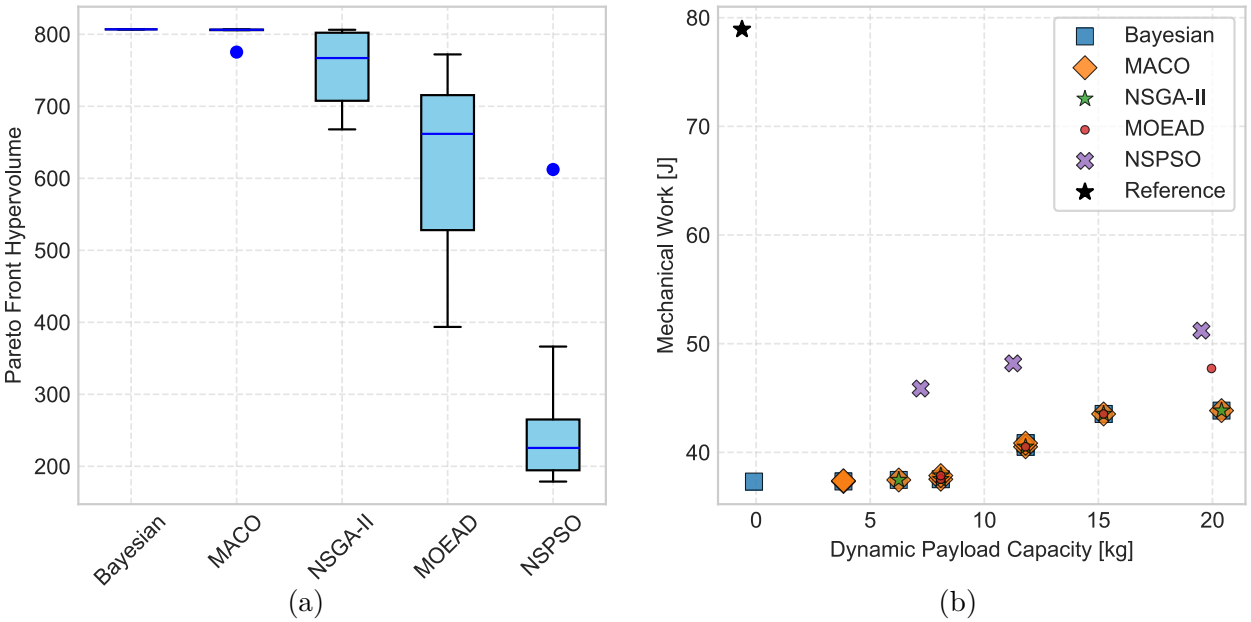


Figure 4.8 Performance of each algorithm on the multi-objective formulation of Problem 3. (a) Distribution of Pareto front hypervolumes across 10 independent runs. (b) Best Pareto front achieved by each algorithm, selected based on the hypervolume criterion.

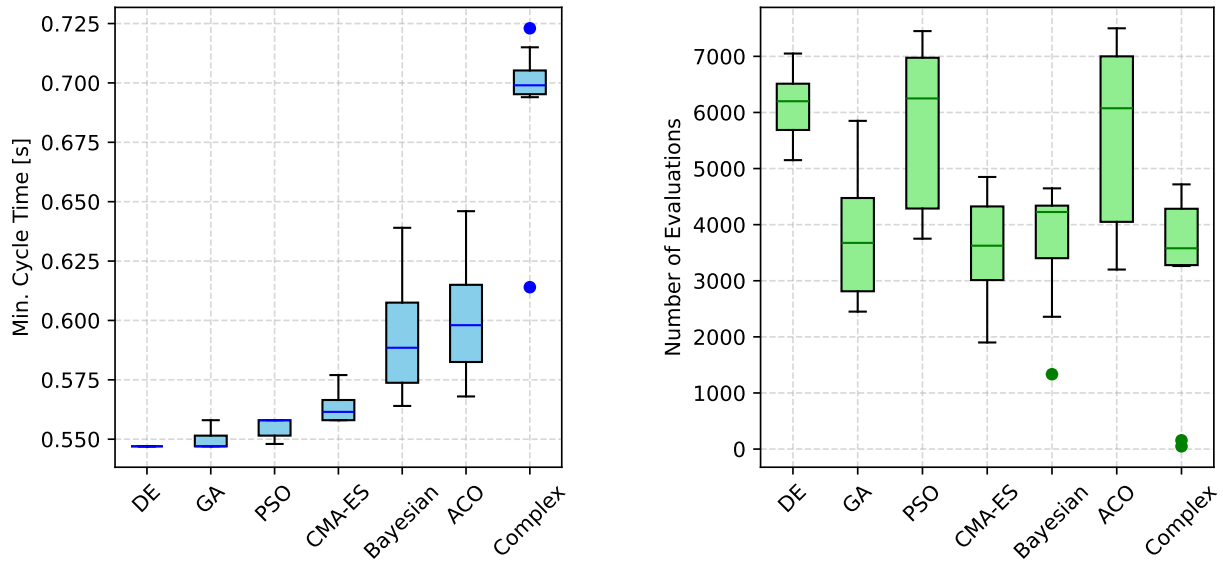
4.3.4 Problem 4

As outlined in Section 4.2.4, Problem 4 is formulated to identify the optimal selection of motors and reducers at each joint in order to minimize the minimum cycle time required to execute Task 2. Figure 4.9 presents box plots showing the distribution of the best objective values and the number of function evaluations required to reach them across ten independent runs for each of the seven evaluated algorithms.

The best design obtained (Table D.4) reduces the cycle time to 0.55 s, a 30% reduction over the original design's 0.78 s.

Due to the high dimensionality of the design space, this problem poses a greater optimization challenge than earlier cases. The best-performing algorithms are DE and GA, both of which successfully identified the same optimized solution. However, only DE was able to consistently find the best design across all independent runs.

Interestingly, the three algorithms that performed best in the low-dimensional discrete case (Problem 1), Bayesian optimization, CMA-ES, and PSO, struggled in this higher-dimensional setting. However, while CMA-ES and PSO still converged to good designs, Bayesian optimization struggled significantly, failing to produce competitive results in the majority of the runs.



(a) (b)

Figure 4.9 Box plots showing the performance of each algorithm over 10 independent runs on Problem 4. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

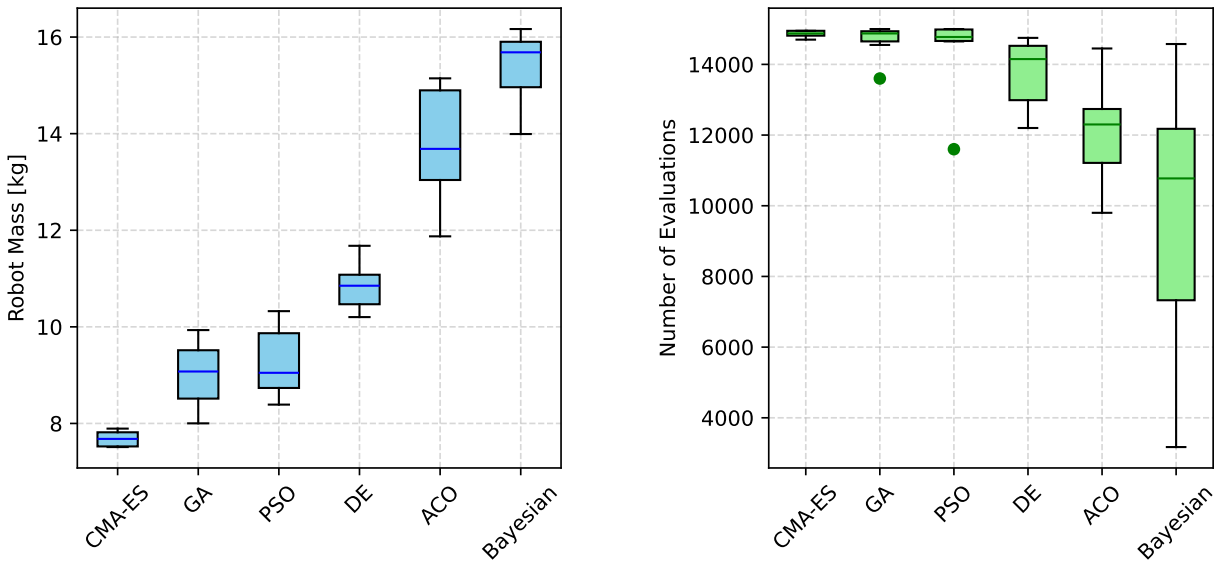
4.3.5 Problem 5

As detailed in Section 4.2.5, Problem 5 seeks to identify the optimal ETS parameters of the robot that minimize its mass while maintaining 100% dexterity at every point described in Task 1. Figure 4.10 presents box plots illustrating the distribution of the best objective values and the number of function evaluations required to reach them across ten independent runs for each of the evaluated algorithms that converged towards feasible designs.

The best designs are consistently obtained with CMA-ES, which identifies feasible solutions under 8 kg in all runs. The best design, with a mass of 7.55 kg, is reported in Table D.5. These results represent an improvement over the reference design, achieving 100% task dexterity compared to the reference's average of 62%, while also reducing the overall design weight by 13%.

The SQP algorithm is excluded from Figure 4.10 as it failed to produce any feasible solutions. This is likely due to the inherently discontinuous nature of the dexterity constraint, which hinders the performance of gradient-based methods relying on finite difference approximations to compute descent directions. The Complex algorithm is also omitted, as none of its runs converged to a design that satisfied the dexterity constraint.

Furthermore, the number of function evaluations required by CMA-ES, GA, and PSO to reach their best solutions approaches the termination threshold of 15,000 evaluations. This suggests that these algorithms might benefit from a relaxed termination criterion, potentially enabling further reductions in mass.



(a) (b)

Figure 4.10 Box plots showing the performance of each algorithm over 10 independent runs on Problem 5. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

The multi-objective formulation introduced in Section 4.2.5 enables the exploration of trade-offs between mass and average dexterity along the task. Figure 4.11 presents the distribution of Pareto front hypervolumes across ten independent runs for each algorithm, along with the best Pareto front obtained by each method, selected based on the hypervolume criterion.

The best-performing algorithm is NSGA-II, which consistently produces dominant Pareto fronts, followed closely by MOEAD and MACO. In contrast, the Bayesian optimization algorithm struggles to identify non-dominated solutions in the high-dimensional design space, and NSPSO exhibits similar limitations as observed in previous problems.

It is worth highlighting that only MOEAD and NSGA-II were able to identify designs with 100% dexterity, corresponding to masses of approximately 9 kg. In comparison, the best design obtained in the mono-objective formulation weighted 7.5 kg. This discrepancy suggests that the Pareto fronts obtained by the algorithms have not yet converged to the true Pareto front. However, it is expected that further convergence could be achieved by relaxing the termination criterion.

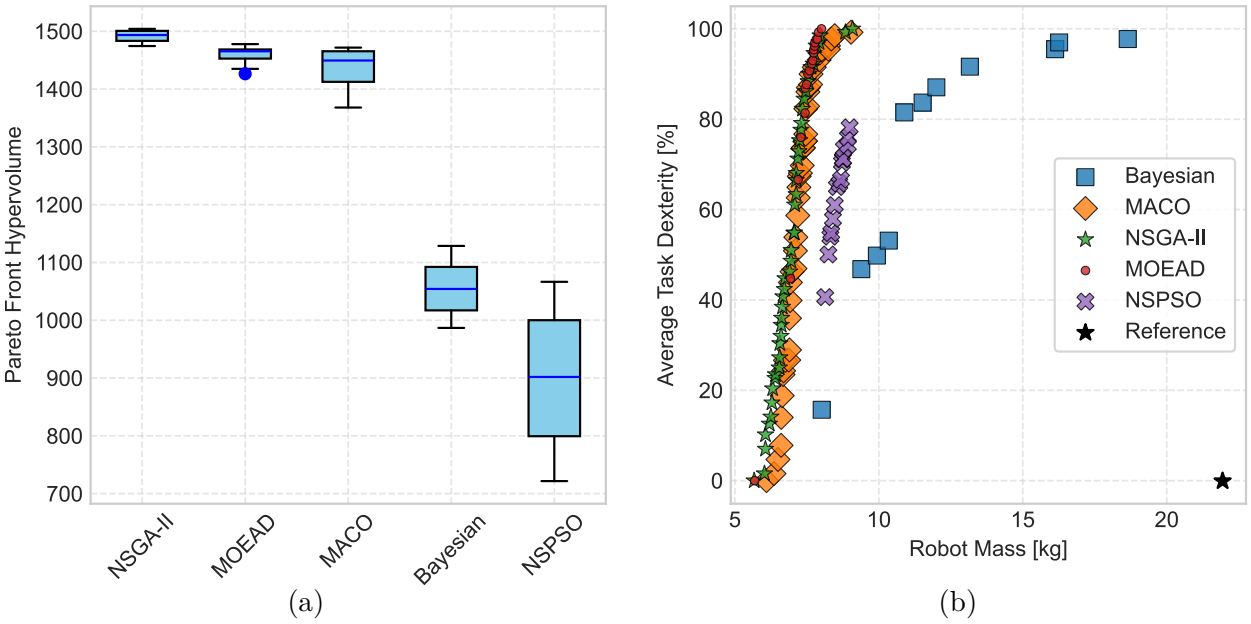


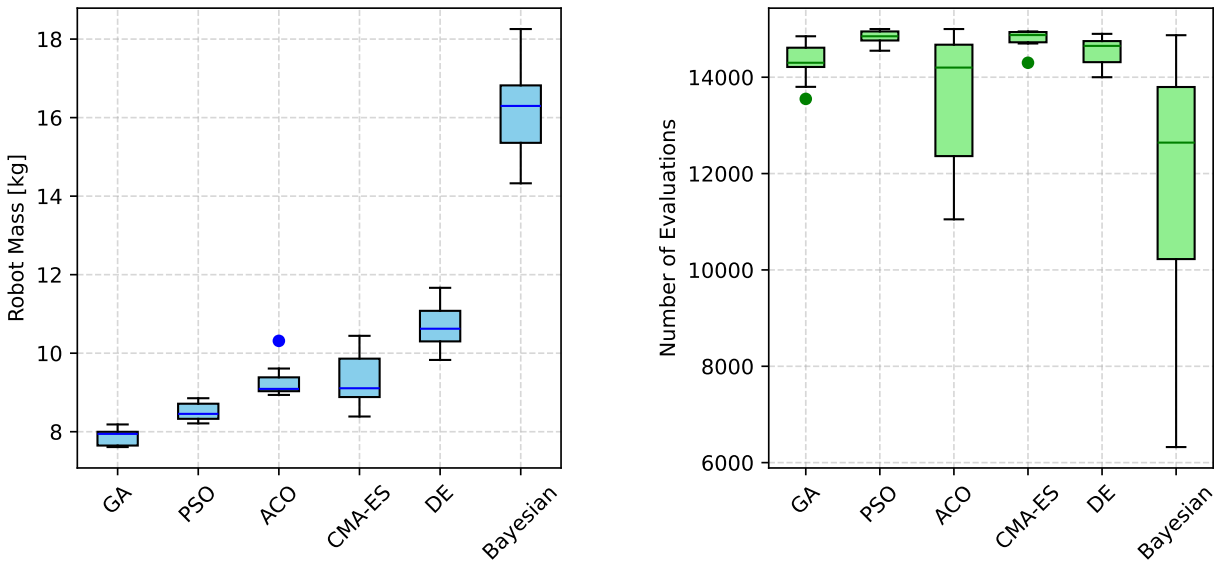
Figure 4.11 Performance of each algorithm on the multi-objective formulation of Problem 5. (a) Distribution of Pareto front hypervolumes across 10 independent runs. (b) Best Pareto front achieved by each algorithm, selected based on the hypervolume criterion.

4.3.6 Problem 6

Finally, Problem 6, presented in Section 4.2.6, aims to identify the optimal combination of link lengths, motors, and reducers to minimize the robot's total mass. The problem is subject to several constraints, including reachability of Task 1, global payload capacity, global manipulability, and dynamic payload capacity on Task 3. Figure 4.12 presents box plots illustrating the distribution of best objective values and the number of function evaluations required to achieve them, based on ten independent runs for each of the evaluated algorithms that converged towards feasible designs.

The best-performing algorithm is consistently the GA, which produced the lightest feasible design with a mass of 7.6 kg, as reported in Table D.6. This design outperforms the reference configuration with a 13% mass reduction and a 220% improvement of dynamic payload capacity on Task 3 , while achieving equivalent global performance in terms of GCI and global payload capacity through the enforced constraints. As with Problem 5, the Complex algorithm is excluded from the results due to its inability to identify any feasible solutions.

Additionally, the tendency of several algorithms to find their best solution toward the end of the allotted 15,000 evaluations suggests that, as in Problem 5, a relaxed termination criterion may allow convergence to improved designs.



(a) (b)

Figure 4.12 Box plots showing the performance of each algorithm over 10 independent runs on Problem 6. (a) Distribution of the best objective values obtained. (b) Number of function evaluations required to reach the best value.

The multi-objective formulation introduced in Section 4.2.6 aims to highlight the trade-offs between dynamic payload capacity on Task 3 and robot mass. Figure 4.13 presents the distribution of Pareto front hypervolumes across ten independent runs for each algorithm, along with the best Pareto front obtained by each method, selected based on the hypervolume criterion.

Although the best Pareto front overall is found with NSGA-II, the distribution shows that MOEAD and MACO perform similarly on average. The most important observation however, is the distribution of results for all algorithms. It hints at the fact that the algorithms may need a more relaxed termination criteria to converge closer towards the true Pareto front. The fact that none of the best Pareto fronts present designs with masses inferior to 10 kg like the design found with the mono-objective formulation tends to confirm this conclusion.

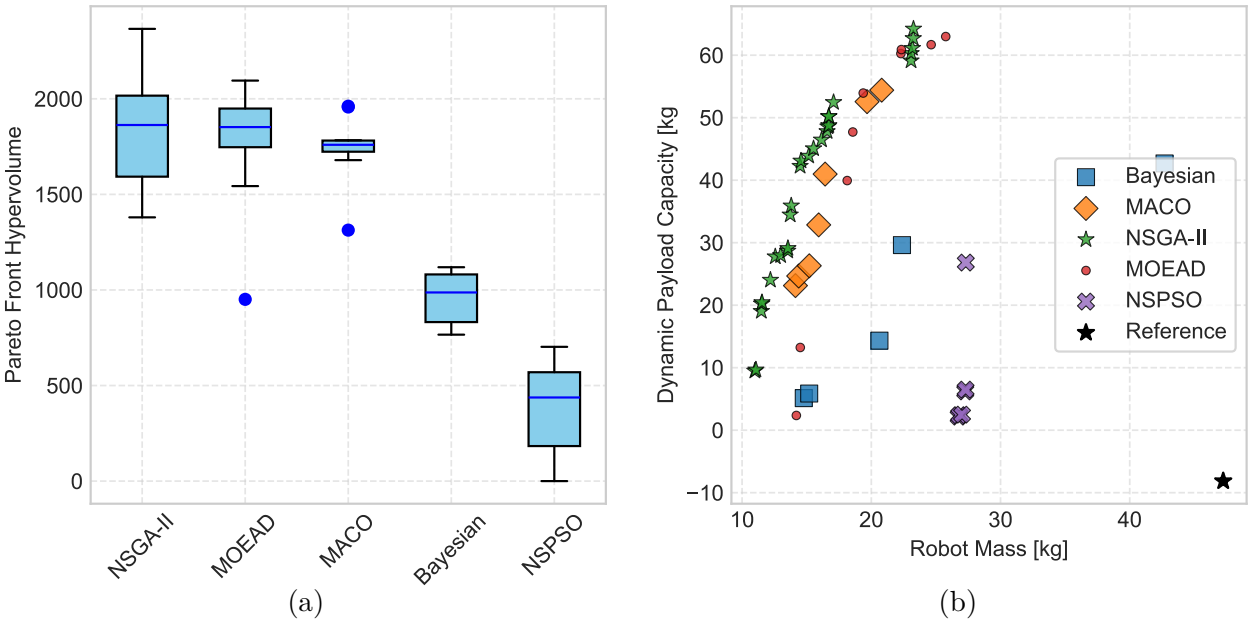


Figure 4.13 Performance of each algorithm on the multi-objective formulation of Problem 6. (a) Distribution of Pareto front hypervolumes across 10 independent runs. (b) Best Pareto front achieved by each algorithm, selected based on the hypervolume criterion.

4.3.7 General Observations

The results obtained across all six problems underscore that algorithmic performance is strongly influenced by problem characteristics.

A first general observation is the strong performance of CMA-ES on mono-objective problems, where it identifies the best design in 67% of cases. It consistently ranks among the top two algorithms for low-dimensional problems (Problems 1–3) and for problems involving only continuous variables (Problems 2 and 5), achieving the best design in 100% of these cases. Moreover, in problems where multiple algorithms consistently converged to the best result, CMA-ES was the second most efficient in terms of function evaluations.

While it does not outperform other methods on high-dimensional problems, it still identifies excellent feasible solutions, only 3 to 15% off the best value of the objective function.

For low-dimensional problems involving discrete variables (Problems 1 and 3), Bayesian optimization proves to be the most efficient, reaching the best design with 20–30% fewer function evaluations compared to CMA-ES.

In contrast, for high-dimensional problems with discrete variables (Problems 4 and 6), the GA demonstrates strong performance, consistently ranking either first (in 33% of the problems)

or second (in 67% of the problems).

The performance of the SQP algorithm, only evaluated on Problems 2 and 5, is also noteworthy. On Problem 2, SQP outperforms all other algorithms by a wide margin, consistently finding the best result alongside with CMA-ES, while requiring 18 times fewer iterations. This highlights its effectiveness and constraint-handling capacity when the problem is continuous and smooth, conditions that are well-suited to gradient-based optimization. However, its performance completely deteriorates on Problem 5, where the presence of an inherently discontinuous constraint renders finite-difference gradient estimation ineffective and prevents convergence to feasible solutions.

As for the gradient-free algorithms, no general tendency is observed in terms of performance with constraints. The weighted penalty method seems to have allowed the algorithms to efficiently navigate the constrained design space.

In the context of multi-objective problems, Bayesian optimization performs best for low-dimensional formulations, consistently producing the largest hyper-volume (Problems 2 and 3), while NSGA-II demonstrates superior performance in higher-dimensional cases consistently ranking first. MOEAD also generates high-quality Pareto fronts across most problems, ranking 2nd in 75% of the problems, with the exception of the low-dimensional discrete case. Conversely, NSPSO is the weakest performer, consistently ranking last across all multi-objective benchmarks.

A general decision tree is proposed in Figure 4.14 to assist in the selection of the suitable algorithm depending on the problem formulation.

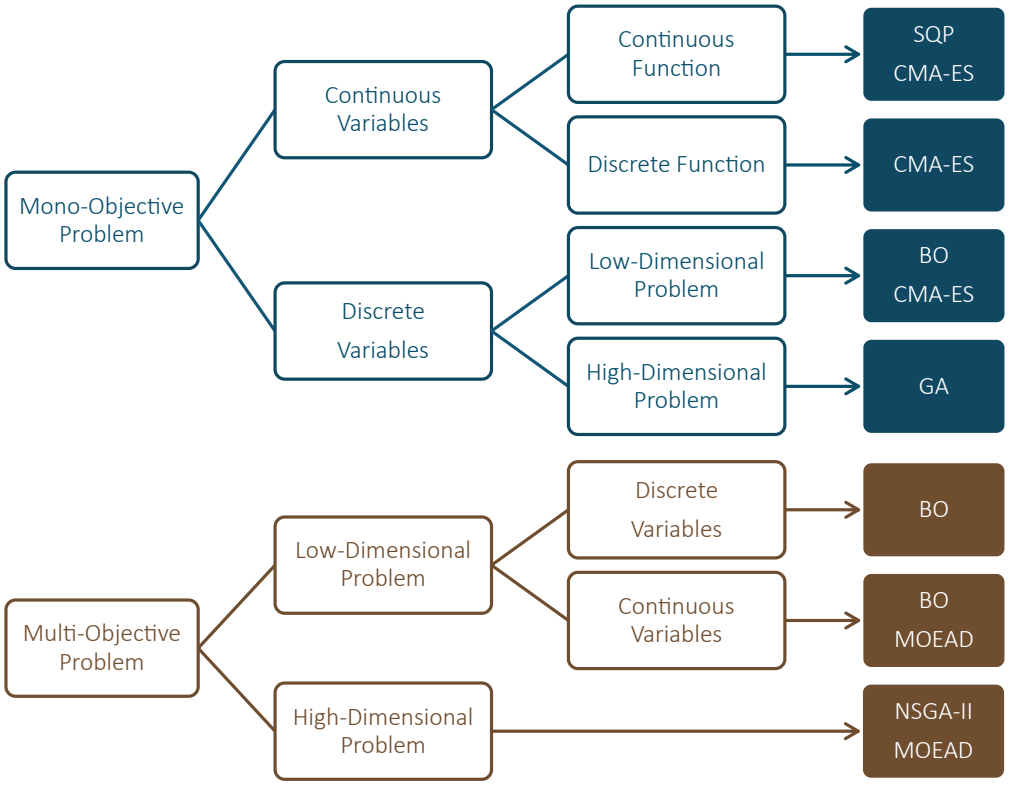


Figure 4.14 Decision tree to assist in selecting a suitable algorithm for manipulator inverse design, based on the problem formulation defined by the number of objective functions, problem dimensionality, and the type of design variables.

To summarize this chapter, the direct design framework was first applied to analyze the reference design in Section 4.1. Then, a set of representative inverse design problems was formulated in Section 4.2. Finally, in Section 4.3, a selection of optimization algorithms was evaluated on each problem to address **SO-3**.

CHAPTER 5 DISCUSSION

The objective of this master's thesis was to develop a versatile framework for the optimization and exploration of conceptual designs for serial robot manipulators. The specific sub-objectives guiding this work are outlined in Section 2.5. This chapter discusses the extent to which each sub-objective has been achieved, identifies the limitations of the developed framework, and proposes directions for future improvement and research.

5.1 Return on the Objectives

SO-1 was addressed through the development of the direct design framework, which introduced a manipulator model capable of representing a wide variety of designs, along with evaluation functions to assess their performance across a broad set of criteria.

The proposed model developed in Section 3.1 integrates kinematic, dynamic, structural, and actuator components into a unified framework, drawing on elements from existing works to create a robust general representation, suitable for conceptual design evaluation. The use of the ETS kinematic representation enhances the interpretability of model parameters, enabling engineers to efficiently parameterize novel designs. The inclusion of an optimized inverse kinematics module allows for the possibility to guide the solver towards preferred configurations when evaluating performance at specific poses or task points. The structural and dynamic models offer a simplified representation of link properties, enabling rapid assessment of design alternatives. Additionally, the actuator model supports a wide range of drivetrain architectures. The inclusion of three motor limits models allows for the evaluation of performance under both static and dynamic conditions, with a variable level of fidelity to be adapted based on the requirements of the evaluation functions.

The performance evaluation functions implemented in Section 3.2 encompass the majority of criteria found in the surveyed literature and commercial data sheets. The four use cases considered enable the analysis of both global performance and task-specific performance, whether in relation to a set of points or poses, a path, or a trajectory. In addition to incorporating established evaluation methods, such as TOPP-RA [126] for minimum cycle time computation, the framework introduces a novel use of optimization techniques to globalize local performance indices by identifying extreme configurations. As demonstrated in Section 4.1, the evaluation process is computationally efficient, with performance assessments completed in under two minutes.

Building on this foundation, **SO-2** was addressed in Section 3.3 through the implementation of the inverse design framework.

This framework enables the selection and optimization of design variables, including both kinematic parameters and actuator configurations. It supports two levels of actuator optimization, either as complete actuator units or as combinations of motors and reducers. This added flexibility allows the framework to better deal with a wide range of application-specific requirements. In doing so, it overcomes a key limitation of existing frameworks [83,124], which support only a restricted range of optimizable parameters.

Furthermore, the ability to formulate inverse design problems using any of the performance evaluation functions from the direct framework enables engineers to tailor the optimization process to a wide array of applications, using both global and task-specific criteria. This level of versatility overcomes another major limitation discussed in Section 2.4, surpassing the Darwin2k framework [123], which is constrained to task-specific metrics.

The results of the design optimization problems presented in Section 4.2 demonstrate the tool's ability to generate improved solutions for various inverse design scenarios, even when starting from a design initially developed for well-balanced performance. A subsequent analysis of the optimized designs using the direct design module would enable engineers to better understand the performance trade-offs implicitly made during the optimization process, thereby helping to assess the relevance and validity of the resulting solutions.

Finally, the inclusion of a variety of optimization algorithms allows the framework to leverage the strengths of different approaches for different problem types, addressing a third major limitation identified in the literature.

To evaluate the effectiveness of the optimization process, **SO-3** was addressed in Sections 4.2 and 4.3 through the introduction of a classification scheme for inverse design problems, the construction of a representative set of such problems, and a comparative analysis of algorithm performance on each case. The results demonstrate the effectiveness of the proposed framework, with optimized designs outperforming the original design across all six studied problems. Additionally, the findings confirm the initial hypothesis that algorithm performance varies depending on the nature and structure of the problem. Based on these observations, general recommendations are proposed to assist in selecting the most appropriate algorithm according to the identified problem class.

As this project was conducted in partnership with KINOVA Robotics, the developed framework was fully implemented in Python and C++, and delivered as a proof of concept to support engineers in responding to requests for proposal.

5.2 Limitations and Directions for Improvement

As discussed in previous chapters, the developed framework has certain limitations in terms of both scope and reliability.

One of the most significant constraints lies in the structural model, which currently represents each link as a hollow cylindrical shell. While this level of abstraction may suffice for early-stage concepts design, it limits the framework's ability to accurately capture the dynamic properties of more complex geometries. Replacing this simplified model with a parameterized CAD-based representation, as proposed by Tarkian et al. [98], would enable more accurate modeling of link geometries at the cost of computational efficiency, and could help establish relationships between shell dimensions and physical joint limits.

Another notable limitation is the exclusive focus on revolute joints, which restricts the exploration of novel architectures that combine different joint types. Future work could extend the framework to include prismatic joints, requiring adjustments to both the manipulator model and performance evaluation functions.

In addition, the current actuator model is limited to electric actuators. Expanding the framework to incorporate other actuator types, such as hydraulic actuators, would significantly broaden its applicability, particularly for large-scale robotic systems.

Another limitation lies in the absence of structural integrity considerations within the implemented evaluation criteria. The framework currently depends on the engineer's expertise to select appropriate shell parameters that minimize deflection and avoid material failure. Future work could address this by incorporating methods such as the lumped parameter approach [113] or finite element analysis [109], enabling the evaluation of stress and deflection in the shell components. These analyses could then be integrated as constraints within inverse design problem formulations. Additionally, the inclusion of obstacle avoidance in the reachability and path planning functions would significantly enhance the framework, allowing for the modeling of more complex and realistic task environments.

Lastly, the recommendations for algorithm selection should be interpreted with caution, as the generalizability of results to other problems in the same class was not assessed. To establish more reliable guidelines, a broader set of test problems should be analyzed to validate the suitability of the proposed classification and to determine whether the observed performance trends extend to other problems within each class. Furthermore, a key limitation of the algorithm comparison methodology is the manual tuning of algorithm hyper-parameters. A more rigorous study on hyper-parameter optimization would allow for a fairer comparison between algorithms by ensuring that each operates under appropriately tuned settings.

CHAPTER 6 CONCLUSION

The work of this thesis has led to the development of a versatile tool to support the conceptual design of serial manipulator, through the creation of a framework including both direct and inverse design approaches.

The proposed framework addresses several key limitations in the existing literature by establishing a generalizable manipulator model that incorporates key considerations from various disciplines. It evaluates manipulator performance across a broad range of applications, leveraging both established metrics and novel evaluation techniques to ensure comprehensive and efficient assessment, with the evaluation of all included criteria on the reference Kinova GEN-3 manipulator model all requiring less than 2 minutes. The inverse design methodology supports the optimization of an extensive set of design parameters, encompassing both kinematic and actuation parameters. Furthermore, the framework provides guidelines for algorithm selection tailored to different classes of design problems, capitalizing on specific algorithmic strengths.

Specifically, the excellent performance of the CMA-ES algorithm on mono-objective problems is highlighted, as it achieves the best result in 67% of the studied problems. Bayesian optimization also demonstrates strong efficiency in low-dimensional problems with discrete variables, requiring 20-30% fewer function evaluations than CMA-ES to reach the best solution. Additionally, Bayesian optimization ranks first among the compared algorithms for low-dimensional multi-objective problems, whereas NSGA-II outperforms the others in higher-dimensional settings.

Results demonstrate the framework's capacity to consistently generate high-performing design solutions, surpassing the reference design across all tested scenarios. It also enables effective trade-off analysis, offering insight into the compromises inherent in multi-objective robotic design. The implementation of the framework in Python for use by the project's industrial partner offers a tangible tool to streamline the early stages of manipulator design and support the exploration of innovative concepts.

Nonetheless, several limitations were identified. These include constraints in the model's representational fidelity, particularly regarding shell modeling, the omission of structural integrity considerations, and the limited generalizability of the current algorithm selection guidelines.

Given the broad scope of the framework, multiple avenues for future research and develop-

ment are apparent. From a modeling standpoint, integration with engineering software such as CAD tools or finite element analysis platforms could enhance the accuracy of performance estimations. The model could also be expanded to incorporate a wider variety of design parameters, joint types, and actuator models. Additionally, further work is warranted to broaden and generalize the performance criteria through the development of additional evaluation functions applicable to any combination of design parameters. The robustness of the algorithm selection guidelines could be improved through larger-scale studies that examine a diverse set of design challenges and optimize the hyper-parameters of competing algorithms. Finally, validation of the framework's performance estimations could be achieved by prototyping an optimized design and experimentally comparing its actual performance to the estimated results.

In conclusion, this research contributes to the growing body of work in manipulator optimization by introducing a comprehensive framework capable of addressing a wide range of design challenges. It provides a foundation for future developments in the field and offers a practical tool for both academic research and industrial application.

REFERENCES

[1] Intuitive. Da Vinci Robotic Surgical Systems. [Online]. Available: https://www.intuitive.com/en-us/products-and-services/da-vinci

[2] Medtronic. Hugo™ RAS System. [Online]. Available: https://www.medtronic.com/covidien/en-ca/robotic-assisted-surgery/hugo-ras-system.html

[3] Kinova Assistive Technologies. Jaco robotic arm. [Online]. Available: https://assistive.kinovarobotics.com/product/jaco-robotic-arm

[4] Canadian Space Agency, "Canadarm," Nov. 2006. [Online]. Available: https://www.asc-csa.gc.ca/eng/canadarm/

[5] J. Xiang, L. Wang, L. Li, K.-H. Lai, and W. Cai, "Classification-design-optimization integrated picking robots: a review," *Journal of Intelligent Manufacturing*, vol. 35, no. 7, pp. 2979–3002, Oct. 2024. [Online]. Available: https://doi.org/10.1007/s10845-023-02201-5

[6] R. Bischoff, U. Huggenberger, and E. Prassler, "KUKA youBot - a mobile manipulator for research and education," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1–4, iSSN: 1050-4729. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5980575

[7] K. Yoshida, B. Wilcox, G. Hirzinger, and R. Lampariello, "Space Robotics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Cham: Springer International Publishing, 2016, pp. 1423–1462. [Online]. Available: https://doi.org/10.1007/978-3-319-32552-1_55

[8] E. M. Hoffman, A. Laurenzi, F. Ruscelli, L. Rossini, L. Baccelliere, D. Antonucci, A. Margan, P. Guria, M. Migliorini, S. Cordasco, G. Raiola, L. Muratore, J. E. Rodrigo, A. Rusconi, G. Sangiovanni, and N. G. Tsagarakis, "Design and Validation of a Multi-Arm Relocatable Manipulator for Space Applications," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 11 887–11 893. [Online]. Available: https://ieeexplore.ieee.org/document/10160389

[9] J. Trevelyan, W. R. Hamel, and S.-C. Kang, "Robotics in Hazardous Applications," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Cham:

Springer International Publishing, 2016, pp. 1521–1548. [Online]. Available: https://doi.org/10.1007/978-3-319-32552-1_58

[10] J. A. Marshall, A. Bonchis, E. Nebot, and S. Scheding, "Robotics in Mining," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Cham: Springer International Publishing, 2016, pp. 1549–1576. [Online]. Available: https://doi.org/10.1007/978-3-319-32552-1_59

[11] T. Kivelä, J. Mattila, and J. Puura, "A generic method to optimize a redundant serial robotic manipulator's structure," *Automation in Construction*, vol. 81, pp. 172–179, Sep. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580517305150

[12] C. Müller, "World robotics 2024 – industrial robots," IFR Statistical Department, VDMA Services GmbH, Frankfurt am Main, Germany, 2024.

[13] M. A. Laribi, G. Carbone, and S. Zeghloul, "Robot Design: Optimization Methods and Task-Based Design," in *Robot Design: From Theory to Service Applications*, G. Carbone and M. A. Laribi, Eds. Cham: Springer International Publishing, 2023, pp. 97–115. [Online]. Available: https://doi.org/10.1007/978-3-031-11128-0_5

[14] E. M. Hoffman, D. Costanzi, G. Fadini, N. Miguel, A. D. Prete, and L. Marchionni, "Addressing Reachability and Discrete Component Selection in Robotic Manipulator Design Through Kineto-Static Bi-Level Optimization," *IEEE Robotics and Automation Letters*, vol. 10, no. 3, pp. 2263–2270, Mar. 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10842359

[15] "Robotics – Vocabulary," ISO 8372, 2021.

[16] M. Ceccarelli, "Fundamentals of the Mechanics of Serial Manipulators," in *Fundamentals of Mechanics of Robotic Manipulation*, M. Ceccarelli, Ed. Cham: Springer International Publishing, 2022, pp. 81–206. [Online]. Available: https://doi.org/10.1007/978-3-030-90848-5_3

[17] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, ser. Springer Handbooks. Cham: Springer International Publishing, 2016. [Online]. Available: https://link.springer.com/10.1007/978-3-319-32552-1

[18] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in Python*, ser. Springer Tracts in Advanced Robotics. Cham: Springer International Publishing, 2023, vol. 146. [Online]. Available: https://link.springer.com/10.1007/978-3-031-06469-2

[19] J. Denavit and R. S. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *Journal of Applied Mechanics*, vol. 22, no. 2, pp. 215–221, Jun. 1955. [Online]. Available: https://asmedigitalcollection.asme.org/appliedmechanics/article/22/2/215/1110292/A-Kinematic-Notation-for-Lower-Pair-Mechanisms

[20] R. W. Brockett, "Robotic manipulators and the product of exponentials formula," in *Mathematical Theory of Networks and Systems*, P. A. Fuhrmann, Ed. Berlin, Heidelberg: Springer, 1984, pp. 120–129.

[21] P. I. Corke, "A Simple and Systematic Approach to Assigning Denavit–Hartenberg Parameters," *IEEE Transactions on Robotics*, vol. 23, no. 3, pp. 590–594, Jun. 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4252158

[22] P. Chang, "A closed-form solution for inverse kinematics of robot manipulators with redundancy," *IEEE Journal on Robotics and Automation*, vol. 3, no. 5, pp. 393–403, Oct. 1987. [Online]. Available: https://ieeexplore.ieee.org/document/1087114

[23] A. Deo and I. Walker, "Adaptive non-linear least squares for inverse kinematics," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, May 1993, pp. 186–193 vol.1. [Online]. Available: https://ieeexplore.ieee.org/document/291981

[24] T. Sugihara, "Solvability-Unconcerned Inverse Kinematics by the Levenberg–Marquardt Method," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 984–991, Oct. 2011. [Online]. Available: https://ieeexplore.ieee.org/document/5784347

[25] C. W. Wampler, "Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, no. 1, pp. 93–101, Jan. 1986. [Online]. Available: https://ieeexplore.ieee.org/document/4075580

[26] S. Chan and P. Lawrence, "General inverse kinematics with the error damped pseudoinverse," in *1988 IEEE International Conference on Robotics and Automation Proceedings*, Apr. 1988, pp. 834–839 vol.2. [Online]. Available: https://ieeexplore.ieee.org/document/12164

[27] J. Haviland and P. Corke, "Manipulator Differential Kinematics: Part I: Kinematics, Velocity, and Applications [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 31, no. 4, pp. 149–158, Dec. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10126113/references#references

[28] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "On-Line Computational Scheme for Mechanical Manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, no. 2, pp. 69–76, Jun. 1980. [Online]. Available: https://doi.org/10.1115/1.3149599

[29] M. Russo, "Measuring Performance: Metrics for Manipulator Design, Control, and Optimization," *Robotics*, vol. 12, no. 1, p. 4, Feb. 2023. [Online]. Available: https://www.mdpi.com/2218-6581/12/1/4

[30] Kinova. Gen3 - ultra-lightweight robotic arm. [Online]. Available: https://www.kinovarobotics.com/product/gen3-robots

[31] KUKA. Lbr iiwa. [Online]. Available: https://www.kuka.com/en-us/products/robotics-systems/industrial-robots/lbr-iiwa

[32] Franka Robotics. Franka research 3. [Online]. Available: https://franka.de/products/franka-research-3

[33] ABB. Articulated robot irb 1300. [Online]. Available: https://new.abb.com/products/robotics/robots/articulated-robots/irb-1300

[34] A. Mohebbi, "Concurrent, integrated and multicriteria design support for mechatronic systems," Ph.D. dissertation, École Polytechnique de Montréal, Mar. 2017. [Online]. Available: https://publications.polymtl.ca/2508/

[35] H. A. Moreno, R. Saltaren, I. Carrera, L. Puglisi, and R. Aracil, "Ìndices de Desempeño de Robots Manipuladores: Una revisión del Estado del Arte," *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 9, no. 2, pp. 111–122, Apr. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1697791212000064

[36] S. Patel and T. Sobh, "Manipulator Performance Measures - A Comprehensive Literature Survey," *Journal of Intelligent & Robotic Systems*, vol. 77, no. 3, pp. 547–570, Mar. 2015. [Online]. Available: https://doi.org/10.1007/s10846-014-0024-y

[37] K. C. Gupta and B. Roth, "Design Considerations for Manipulator Workspace," *Journal of Mechanical Design*, vol. 104, no. 4, pp. 704–711, Oct. 1982. [Online]. Available: https://doi.org/10.1115/1.3256412

[38] M. Ceccarelli, "Fundamentals of the Mechanics of Serial Manipulators," in *Fundamentals of Mechanics of Robotic Manipulation*, M. Ceccarelli, Ed. Cham:

Springer International Publishing, 2022, pp. 81–206. [Online]. Available: https://doi.org/10.1007/978-3-030-90848-5_3

[39] ——, "Challenges for workspace analysis of manipulators," *Journal of the Indian Institute of Science*, Feb. 2025. [Online]. Available: https://doi.org/10.1007/s41745-024-00458-0

[40] C. J. J. Paredis and P. K. Khosla, "Kinematic Design of Serial Link Manipulators From Task Specifications," *The International Journal of Robotics Research*, vol. 12, no. 3, pp. 274–287, Jun. 1993. [Online]. Available: https://doi.org/10.1177/027836499301200306

[41] S. Kucuk and Z. Bingul, "Comparative study of performance indices for fundamental robot manipulators," *Robotics and Autonomous Systems*, vol. 54, no. 7, pp. 567–573, Jul. 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889006000546

[42] E. J. Munoz Montenegro, M. S. Haque Sunny, J. D. S. De Caro, B. Brahmi, J. Ghommam, M. Saad, H. U. Ahmed, and M. H. Rahman, "Kinematic Optimization and Comparison of Wheelchair-mounted Assistive Robots for Activities of Daily Living," in *2023 IEEE 14th International Conference on Power Electronics and Drive Systems (PEDS)*, Aug. 2023, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/10246520

[43] S. Patel and T. Sobh, "Goal directed design of serial robotic manipulators," in *Proceedings of the 2014 Zone 1 Conference of the American Society for Engineering Education*, Apr. 2014, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/6820684

[44] J. D. Sanjuan De Caro, M. S. H. Sunny, E. Muñoz, J. Hernandez, A. Torres, B. Brahmi, I. Wang, J. Ghommam, and M. H. Rahman, "Evaluation of Objective Functions for the Optimal Design of an Assistive Robot," *Micromachines*, vol. 13, no. 12, p. 2206, Dec. 2022. [Online]. Available: https://www.mdpi.com/2072-666X/13/12/2206

[45] K. Kawaharazuka, T. Makabe, K. Okada, and M. Inaba, "Daily Assistive Modular Robot Design Based on Multi-Objective Black-Box Optimization," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2023, pp. 9970–9977. [Online]. Available: https://ieeexplore.ieee.org/document/10342041

[46] K. Kawaharazuka, K. Okada, and M. Inaba, "Robot Design Optimization with Rotational and Prismatic Joints using Black-Box Multi-Objective Optimization," in

2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Oct. 2024, pp. 4571–4577. [Online]. Available: https://ieeexplore.ieee.org/document/10802642

[47] O. Chocron and P. Bidaud, "Evolutionary algorithms in kinematic design of robotic systems," in *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97*, vol. 2, Sep. 1997, pp. 1111–1117 vol.2. [Online]. Available: https://ieeexplore.ieee.org/document/655148

[48] C. Baykal, C. Bowen, and R. Alterovitz, "Asymptotically optimal kinematic design of robots using motion planning," *Autonomous Robots*, vol. 43, no. 2, pp. 345–357, Feb. 2019. [Online]. Available: https://doi.org/10.1007/s10514-018-9766-x

[49] X. Zhu, P. Gergondet, Z. Cai, X. Chen, Z. Yu, Q. Huang, and A. Kheddar, "The Development of a 7-DoF Humanoid Arm for Driving Using a Task-Driven Design Method," *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 2, pp. 1521–1533, Apr. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10250896

[50] X. Zhu, H. Liu, Q. Li, Z. Cai, X. Chen, Z. Yu, Q. Huang, and A. Kheddar, "Structural Analysis and Design of Humanoid Arms From Human Arm Reachable Workspace," *IEEE Robotics and Automation Letters*, vol. 10, no. 2, pp. 1233–1240, Feb. 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10806768

[51] L. J. Puglisi, R. J. Saltaren, H. A. Moreno, P. F. Cárdenas, C. Garcia, and R. Aracil, "Dimensional synthesis of a spherical parallel manipulator based on the evaluation of global performance indexes," *Robotics and Autonomous Systems*, vol. 60, no. 8, pp. 1037–1045, Aug. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889012000693

[52] W. Wang, W. Wang, W. Dong, H. Yu, Z. Yan, and Z. Du, "Dimensional optimization of a minimally invasive surgical robot system based on NSGA-II algorithm," *Advances in Mechanical Engineering*, vol. 7, no. 2, p. 1687814014568541, Jan. 2015. [Online]. Available: https://doi.org/10.1177/1687814014568541

[53] J. Li, J. Liu, Y. Hu, H. Ding, and J. Pang, "Integrated Optimization for Service Robotic Arms Involving Workspace, Drive Train, Structural Stiffness and Lightweight," in *2021 5th International Conference on Robotics and Automation Sciences (ICRAS)*, Jun. 2021, pp. 44–50. [Online]. Available: https://ieeexplore.ieee.org/document/9476411

[54] T. Tanev and B. Stoyanov, "On the Performance Indexes for Robot Manipulators," 2000. [Online]. Available: https://www.semanticscholar.org/paper/On-the-Performance-Indexes-for-Robot-Manipulators-Tanev-Stoyanov/699ab4068b4acfed9b89a89c6366f0d0f0ba3dbe

[55] B. Cao, K. Sun, Y. Gu, M. Jin, and H. Liu, "Workspace Analysis Based on Manipulator Pose Dexterity Map," in *2018 3rd International Conference on Robotics and Automation Engineering (ICRAE)*, Nov. 2018, pp. 166–170. [Online]. Available: https://ieeexplore.ieee.org/document/8586768

[56] K. Leibrandt, L. da Cruz, and C. Bergeles, "Designing Robots for Reachability and Dexterity: Continuum Surgical Robots as a Pretext Application," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2989–3007, Aug. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10149817

[57] T. Yoshikawa, "Manipulability of Robotic Mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, Jun. 1985. [Online]. Available: https://doi.org/10.1177/027836498500400201

[58] J.-O. Kim and K. Khosla, "Dexterity measures for design and control of manipulators," in *Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*, Nov. 1991, pp. 758–763 vol.2. [Online]. Available: https://ieeexplore.ieee.org/document/174572

[59] C. Gosselin and J. Angeles, "A Global Performance Index for the Kinematic Optimization of Robotic Manipulators," *Journal of Mechanical Design*, vol. 113, no. 3, pp. 220–226, Sep. 1991. [Online]. Available: https://doi.org/10.1115/1.2912772

[60] T. Yoshikawa, "Dynamic manipulability of robot manipulators," in *1985 IEEE International Conference on Robotics and Automation Proceedings*, vol. 2, Mar. 1985, pp. 1033–1038. [Online]. Available: https://ieeexplore.ieee.org/document/1087277

[61] O. Ma and J. Angeles, "The concept of dynamic isotropy and its applications to inverse kinematics and trajectory planning," in *, IEEE International Conference on Robotics and Automation Proceedings*, May 1990, pp. 481–486 vol.1. [Online]. Available: https://ieeexplore.ieee.org/document/126024

[62] "Manipulating Industrial Robots – Performance Criteria and Related Test Methods," ISO 9283, 1998.

[63] J. Angeles and F. C. Park, "Performance Evaluation and Design Criteria," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2008, pp. 229–244. [Online]. Available: https://doi.org/10.1007/978-3-540-30301-5_11

[64] Q. Fan, Z. Gong, B. Tao, Y. Gao, Z. Yin, and H. Ding, "Base position optimization of mobile manipulators for machining large complex components," *Robotics and Computer-Integrated Manufacturing*, vol. 70, p. 102138, Aug. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0736584521000235

[65] Q. Xu, Q. Zhan, and X. Tian, "Link Lengths Optimization Based on Multiple Performance Indexes of Anthropomorphic Manipulators," *IEEE Access*, vol. 9, pp. 20 089–20 099, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9335994

[66] C. M. Gosselin, "The optimum design of robotic manipulators using dexterity indices," *Robotics and Autonomous Systems*, vol. 9, no. 4, pp. 213–226, 1992. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0921889092900392

[67] J. Külz and M. Althoff, "Optimizing Modular Robot Composition: A Lexicographic Genetic Algorithm Approach," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 16 752–16 758. [Online]. Available: https://ieeexplore.ieee.org/document/10609979

[68] S. Ha, S. Coros, A. Alspach, J. M. Bern, J. Kim, and K. Yamane, "Computational Design of Robotic Devices From High-Level Motion Specifications," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1240–1251, Oct. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8395009

[69] M. Ceccarelli and C. Lanni, "A multi-objective optimum design of general 3R manipulators for prescribed workspace limits," *Mechanism and Machine Theory*, vol. 39, no. 2, pp. 119–132, Feb. 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X03001095

[70] P. R. Bergamaschi, S. d. F. P. Saramago, and L. d. S. Coelho, "Comparative study of SQP and metaheuristics for robotic manipulator design," *Applied Numerical Mathematics*, vol. 58, no. 9, pp. 1396–1412, Sep. 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168927407001353

[71] S. Panda, D. Mishra, and B. Biswal, "Revolute manipulator workspace optimization: A comparative study," *Applied Soft Computing*, vol. 13, no. 2, pp. 899–910, Feb. 2013. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1568494612004255

[72] E. Galan-Uribe and L. Morales-Velazquez, "Kinematic Optimization of 6DOF Serial Robot Arms by Bio-Inspired Algorithms," *IEEE Access*, vol. 10, pp. 110 485–110 496, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9919847

[73] Y. Huang, Z. Li, K. Xing, and H. Gong, "A manipulator size optimization method based on dexterous workspace volume," *Cobot*, vol. 1, p. 3, Jan. 2022. [Online]. Available: https://collaborative-robot.org/articles/1-3/v1

[74] W. Chung, J. Han, Y. Youm, and S. Kim, "Task based design of modular robot manipulator using efficient genetic algorithm," in *Proceedings of International Conference on Robotics and Automation*, vol. 1, Apr. 1997, pp. 507–512 vol.1. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/620087

[75] V. Kumar, S. Sen, S. S. Roy, C. Har, and S. Shome, "Design Optimization of Serial Link Redundant Manipulator: An Approach Using Global Performance Metric," *Procedia Technology*, vol. 14, pp. 43–50, 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212017314000437

[76] S. Patel and T. Sobh, "Task based synthesis of serial manipulators," *Journal of Advanced Research*, vol. 6, no. 3, pp. 479–492, May 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090123214001532

[77] S. Tarek and T. Daniel, "Kinematic synthesis of robotic manipulators from task descriptions," in *Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003.*, vol. 2, Jun. 2003, pp. 1018–1023 vol.2. [Online]. Available: https://ieeexplore.ieee.org/document/1223150

[78] O. Chocron, "Evolutionary design of modular robotic arms," *Robotica*, vol. 26, no. 3, pp. 323–330, May 2008. [Online]. Available: https://www.cambridge.org/core/journals/robotica/article/evolutionary-design-of-modular-robotic-arms/FC29BA7042B35217639A88CC8E0A8919

[79] P. Franceschi, S. Mutti, and N. Pedrocchi, "Optimal design of robotic work-cell through hierarchical manipulability maximization," *Robotics and Computer-Integrated Manufacturing*, vol. 78, p. 102401, Dec. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0736584522000886

[80] S. Kucuk and Z. Bingul, "Robot Workspace Optimization Based on a Novel Local and Global Performance Indices," in *Proceedings of the IEEE International Symposium on*

Industrial Electronics, 2005. ISIE 2005., vol. 4, Jun. 2005, pp. 1593–1598. [Online]. Available: https://ieeexplore.ieee.org/document/1529170

[81] H. Lim, S. Hwang, K. Shin, and C. Han, "Comparative study of optimization technique for the global performance indices of the robot manipulator based on an approximate model," *International Journal of Control, Automation and Systems*, vol. 10, no. 2, pp. 374–382, Apr. 2012. [Online]. Available: https://doi.org/10.1007/s12555-012-0217-8

[82] H.-G. Kim, K.-S. Shin, S.-W. Hwang, and C.-S. Han, "Link length determination method for the reduction of the performance deviation of the manipulator: Extension of the valid workspace," *International Journal of Precision Engineering and Manufacturing*, vol. 15, no. 9, pp. 1831–1838, Sep. 2014. [Online]. Available: https://doi.org/10.1007/s12541-014-0536-1

[83] F. Cursi, W. Bai, E. M. Yeatman, and P. Kormushev, "GlobDesOpt: A Global Optimization Framework for Optimal Robot Manipulator Design," *IEEE Access*, vol. 10, pp. 5012–5023, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9674897

[84] S. Kucuk and Z. Bingul, "Link Mass Optimization of Serial Robot Manipulators Using Genetic Algorithm," in *Knowledge-Based Intelligent Information and Engineering Systems*, B. Gabrys, R. J. Howlett, and L. C. Jain, Eds. Berlin, Heidelberg: Springer, 2006, pp. 138–144.

[85] V. Gupta, S. K. Saha, and H. Chaudhary, "Optimum Design of Serial Robots," *Journal of Mechanical Design*, vol. 141, no. 082303, Apr. 2019. [Online]. Available: https://doi.org/10.1115/1.4042623

[86] A. Dogra, S. Sekhar Padhee, and E. Singla, "Optimal Synthesis of Unconventional Links for Modular Reconfigurable Manipulators," *Journal of Mechanical Design*, vol. 144, no. 083304, Apr. 2022. [Online]. Available: https://doi.org/10.1115/1.4054336

[87] M. Toussaint, J.-S. Ha, and O. S. Oguz, "Co-Optimizing Robot, Environment, and Tool Design via Joint Manipulation Planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 6600–6606. [Online]. Available: https://ieeexplore.ieee.org/document/9561256

[88] G. Fadini, T. Flayols, A. D. Prete, and P. Souères, "Simulation Aided Co-Design for Robust Robot Optimization," *IEEE Robotics and Automation*

Letters, vol. 7, no. 4, pp. 11 306–11 313, Oct. 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9863656

[89] H. Al-Dois, A. K. Jha, and R. B. Mishra, "Task-based design optimization of serial robot manipulators," *Engineering Optimization*, vol. 45, no. 6, pp. 647–658, Jun. 2013. [Online]. Available: https://doi.org/10.1080/0305215X.2012.704027

[90] A. Sathuluri, A. V. Sureshbabu, and M. Zimmermann, "Robust co-design of robots via cascaded optimisation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 11 280–11 286. [Online]. Available: https://ieeexplore.ieee.org/document/10161134

[91] G. Bravo-Palacios, A. D. Prete, and P. M. Wensing, "One Robot for Many Tasks: Versatile Co-Design Through Stochastic Programming," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1680–1687, Apr. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8972465

[92] L. Zhou, S. Bai, and M. R. Hansen, "Design optimization on the drive train of a light-weight robotic arm," *Mechatronics*, vol. 21, no. 3, pp. 560–569, Apr. 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957415811000286

[93] P. Chedmail and M. Gautier, "Optimum Choice of Robot Actuators," *Journal of Engineering for Industry*, vol. 112, no. 4, pp. 361–367, Nov. 1990. [Online]. Available: https://doi.org/10.1115/1.2899600

[94] M. Pettersson, P. Krus, and J. Andersson, "On optimal drive train design in industrial robots," in *2005 IEEE International Conference on Industrial Technology*, Dec. 2005, pp. 254–259. [Online]. Available: https://ieeexplore.ieee.org/document/1600645

[95] B. Johansson, M. Pettersson, and J. O¨lvander, "A Component Based Optimization Approach for Modular Robot Design," in *ASME 2007 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, May 2009, pp. 911–920. [Online]. Available: https://dx.doi.org/10.1115/DETC2007-35329

[96] M. Pettersson and J. Ölvander, "Drive Train Optimization for Industrial Robots," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1419–1424, Dec. 2009. [Online]. Available: https://ieeexplore.ieee.org/document/5229231

[97] P. Li, Z. Nie, Z. Li, and X. Liu, "Optimal Design of the Modular Joint Drive Train for Enhancing Cobot Load Capacity and Dynamic Performance," *Chinese Journal*

of Mechanical Engineering, vol. 37, no. 1, p. 57, Jun. 2024. [Online]. Available: https://doi.org/10.1186/s10033-024-01041-5

[98] M. Tarkian, J. Persson, J. Ölvander, and X. Feng, "Multidisciplinary Design Optimization of Modular Industrial Robots by Utilizing High Level CAD Templates," *Journal of Mechanical Design*, vol. 134, no. 124502, Oct. 2012. [Online]. Available: https://doi.org/10.1115/1.4007697

[99] E. A. Padilla-Garcia, C. A. Cruz-Villar, A. Rodriguez-Angeles, and M. A. Moreno-Armendáriz, "Concurrent optimization on the powertrain of robot manipulators for optimal motor selection and control in a point-to-point trajectory planning," *Advances in Mechanical Engineering*, vol. 9, no. 12, p. 1687814017747368, Dec. 2017. [Online]. Available: https://doi.org/10.1177/1687814017747368

[100] E. A. Padilla-Garc´ıa, C. A. Cruz-Villar, and A. Rodriguez-Angeles, "Multi-Objective Design/Control Optimization on the Power Train of Robot Manipulators using a Genetic Algorithm," *Proceedings of the 14th IFToMM World Congress*, pp. 411–420, Oct. 2015. [Online]. Available: https://www.airitilibrary.com/Article/Detail/P20150909001-201510-201511020032-201511020032-411-420

[101] L. Ge, J. Chen, R. Li, and P. Liang, "Optimization design of drive system for industrial robots based on dynamic performance," *Industrial Robot: An International Journal*, vol. 44, no. 6, pp. 765–775, 2017-10-16Z. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/ir-10-2016-0251/full/html

[102] Y. Xiao, Z. Fan, W. Li, S. Chen, L. Zhao, and H. Xie, "A Manipulator Design Optimization Based on Constrained Multi-objective Evolutionary Algorithms," in *2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, Dec. 2016, pp. 199–205. [Online]. Available: https://ieeexplore.ieee.org/document/7823523

[103] A. Jafari, M. Safavi, and A. Fadaei, "A Genetic Algorithm to Optimum Dynamic Performance of Industrial Robots in the Conceptual Design Phase," in *2007 IEEE 10th International Conference on Rehabilitation Robotics*, Jun. 2007, pp. 1129–1135. [Online]. Available: https://ieeexplore.ieee.org/document/4428565

[104] S. Hwang, H. Kim, Y. Choi, K. Shin, and C. Han, "Design optimization method for 7 DOF robot manipulator using performance indices," *International Journal of Precision Engineering and Manufacturing*, vol. 18, no. 3, pp. 293–299, Mar. 2017. [Online]. Available: https://doi.org/10.1007/s12541-017-0037-0

[105] J. Li, L. Zhou, D. Liu, Y. Li, and Z. Song, "An integrated configuration optimization approach for 6-dof serial manipulators on performance indices," in *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, Jul. 2019, pp. 53–58. [Online]. Available: https://ieeexplore.ieee.org/document/9066462/references

[106] L. Zhou, S. Bai, and M. R. Hansen, "Integrated dimensional and drive-train design optimization of a light-weight anthropomorphic arm," *Robotics and Autonomous Systems*, vol. 60, no. 1, pp. 113–122, Jan. 2012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0921889011001771

[107] J. Li, J. Liu, H. Ding, Y. Hu, and J. Pang, "Optimization Design of Configuration, Structure and Drive Train Synthesis for Serial Robotic Arms," in *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, Jul. 2021, pp. 945–950. [Online]. Available: https://ieeexplore.ieee.org/document/9517620

[108] R. Saravanan, S. Ramabalan, N. G. R. Ebenezer, and R. Natarajan, "Evolutionary bi-criteria optimum design of robots based on task specifications," *The International Journal of Advanced Manufacturing Technology*, vol. 41, no. 3, pp. 386–406, Mar. 2009. [Online]. Available: https://doi.org/10.1007/s00170-008-1483-8

[109] L. Zhou and S. Bai, "A New Approach to Design of a Lightweight Anthropomorphic Arm for Service Applications," *Journal of Mechanisms and Robotics*, vol. 7, no. 031001, Aug. 2015. [Online]. Available: https://doi.org/10.1115/1.4028292

[110] C. Castejón, G. Carbone, J. C. G. Prada, and M. Ceccarelli, "A Multi-Objective Optimization of a Robotic Arm for Service Tasks," *Journal of Mechanical Engineering*, 2010.

[111] X. Wang, D. Zhang, C. Zhao, P. Zhang, Y. Zhang, and Y. Cai, "Optimal design of lightweight serial robots by integrating topology optimization and parametric system optimization," *Mechanism and Machine Theory*, vol. 132, pp. 48–65, Feb. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X18315222

[112] M. Hu, H. Wang, and X. Pan, "Multi-objective global optimum design of collaborative robots," *Structural and Multidisciplinary Optimization*, vol. 62, no. 3, pp. 1547–1561, Sep. 2020. [Online]. Available: https://doi.org/10.1007/s00158-020-02563-x

[113] M. O. Gulec and S. Ertugrul, "Pareto front generation for integrated drive-train and structural optimisation of a robot manipulator conceptual design via NSGA-II," *Advances in Mechanical Engineering*, vol. 15, no. 3, p. 16878132231163051, Mar. 2023. [Online]. Available: https://doi.org/10.1177/16878132231163051

[114] P. Shiakolas, D. Koladiya, and J. Kebrle, "Optimum Robot Design Based on Task Specifications Using Evolutionary Techniques and Kinematic, Dynamic, and Structural Constraints," *Inverse Problems in Engineering*, vol. 10, no. 4, pp. 359–375, Jan. 2002. [Online]. Available: https://doi.org/10.1080/1068276021000004706

[115] Z.-j. Du, Y.-q. Xiao, and W. Dong, "Method for optimizing manipulator's geometrical parameters and selecting reducers," *Journal of Central South University*, vol. 20, no. 5, pp. 1235–1244, May 2013. [Online]. Available: https://doi.org/10.1007/s11771-013-1607-7

[116] Y. Tian, H. Wang, X. Pan, and M. Hu, "Configuration Analysis and optimization of Collaborative Robots," in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Mar. 2019, pp. 1319–1324. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8729248

[117] E. Icer, H. A. Hassan, K. El-Ayat, and M. Althoff, "Evolutionary cost-optimal composition synthesis of modular robots considering a given task," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 3562–3568. [Online]. Available: https://ieeexplore.ieee.org/document/8206201

[118] A. Tremblay and L. Baron, "Geometrical synthesis of star-like topology parallel manipulators with a genetic algorithm," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 3, May 1999, pp. 2446–2451 vol.3. [Online]. Available: https://ieeexplore.ieee.org/document/770472

[119] B. K. Rout and R. K. Mittal, "Optimal design of manipulator parameter using evolutionary optimization techniques," *Robotica*, vol. 28, no. 3, pp. 381–395, May 2010. [Online]. Available: https://www.cambridge.org/core/journals/robotica/article/optimal-design-of-manipulator-parameter-using-evolutionary-optimization-techniques/A19D31C4CDD2F80EEF5CEE5480F8D9D2

[120] M. Pettersson, J. Andersson, and P. Krus, "Methods for Discrete Design Optimization," in *ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of

Mechanical Engineers Digital Collection, Jun. 2008, pp. 295–303. [Online]. Available: https://dx.doi.org/10.1115/DETC2005-85202

[121] M. Tarkian, J. O¨lvander, X. Feng, and M. Petterson, "Design Automation of Modular Industrial Robots," in *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, Jul. 2010, pp. 655–664. [Online]. Available: https://dx.doi.org/10.1115/DETC2009-87271

[122] H. Yin, S. Huang, M. He, and J. Li, "A unified design for lightweight robotic arms based on unified description of structure and drive trains," *International Journal of Advanced Robotic Systems*, vol. 14, no. 4, p. 1729881417716383, Jul. 2017. [Online]. Available: https://doi.org/10.1177/1729881417716383

[123] P. C. Leger, "Automated synthesis and optimization of robot configurations: An evolutionary approach," Ph.D. dissertation, Carnegie Mellon University, United States – Pennsylvania, 1999. [Online]. Available: https://www.proquest.com/docview/304499570/abstract/F3DEFA6DD0E4DF2PQ/1

[124] J. Külz, M. Mayer, and M. Althoff, "Timor Python: A Toolbox for Industrial Modular Robotics," Sep. 2023. [Online]. Available: http://arxiv.org/abs/2209.06758

[125] P. Corke and J. Haviland, "Not your grandmother's toolbox – the Robotics Toolbox reinvented for Python," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 11 357–11 363. [Online]. Available: https://ieeexplore.ieee.org/document/9561366

[126] H. Pham and Q.-C. Pham, "A New Approach to Time-Optimal Path Parameterization Based on Reachability Analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, Jun. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8338417

[127] F. Biscani and D. Izzo, "A parallel global multiobjective framework for optimization: pagmo," *Journal of Open Source Software*, vol. 5, no. 53, p. 2338, 2020. [Online]. Available: https://doi.org/10.21105/joss.02338

[128] J. R. R. A. Martins and A. Ning, *Engineering Design Optimization*. Cambridge, UK: Cambridge University Press, January 2022. [Online]. Available: https://mdobook.github.io

[129] N. Hansen, "The CMA Evolution Strategy: A Tutorial," Mar. 2023. [Online]. Available: http://arxiv.org/abs/1604.00772

[130] Y. Ozaki, Y. Tanigaki, S. Watanabe, M. Nomura, and M. Onishi, "Multiobjective Tree-Structured Parzen Estimator," *Journal of Artificial Intelligence Research*, vol. 73, pp. 1209–1250, Apr. 2022. [Online]. Available: https://www.jair.org/index.php/jair/article/view/13188

[131] J. R. R. A. Martins and A. Ning, "Engineering Design Optimization," Nov. 2021. [Online]. Available: https://www.cambridge.org/highereducation/books/engineering-design-optimization/B1B23D00AF79E45502C4649A0E43135B

[132] A. P. Guerreiro, C. M. Fonseca, and L. Paquete, "The Hypervolume Indicator: Problems and Algorithms," May 2020. [Online]. Available: http://arxiv.org/abs/2005.00515

APPENDIX A PARAMETRIZATION OF THE MODEL OF A KINOVA GEN-3 MANIPULATOR

Table A.1 Complete robot model parametrization for the Kinova GEN-3 arm[1]

link	joint	ETS: from proximal end to distal end	limits	actuator	t_{shell}	D_{shell}
`Base link`	1	$\texttt{ETS}(0.00, 0.00, 0.16; -180°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_1)$	$-\infty; +\infty$	Medium	6mm	90mm
`Shoulder link`	2	$\texttt{ETS}(0.00, 0.00, -0.13; 90°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_2)$	$-128°; 128°$	Medium	6mm	90mm
`Bicep link`	3	$\texttt{ETS}(0.00, -0.41, 0.00; -180°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_3)$	$-147°; 147°$	Medium	6mm	90mm
`Forearm link`	4	$\texttt{ETS}(0.00, 0.21, 0.00; 90°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_4)$	$-\infty; \infty$	Small	5mm	80mm
`Wrist link 1`	5	$\texttt{ETS}(0.00, 0.00, -0.11; -90°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_5)$	$-120°; 120°$	Small	4mm	70mm
`Wrist link 2`	6	$\texttt{ETS}(0.00, 0.11, 0.00; 90°, 0°, 0°) \oplus \texttt{R}_\texttt{z}(\texttt{q}_6)$	$-\infty; \infty$	Small	4mm	70mm
`Wrist link 3`	-	$\texttt{ETS}(0.00, 0.00, -0.06; 0°, 0°, 0°)$	-	-	3mm	70mm

[1] All parameters except from shell thickness and diameters are extracted from the official URDF file.

APPENDIX B EXAMPLE ACTUATOR CATALOG

Table B.1 Example actuator catalog with corresponding model parameter

Component	Parameter	Small	Medium	Large	XLarge
Motor					
	Mass (kg)	0.076	0.135	0.162	0.356
	Inertia ($\times 10^{-6}$ kg·m^2)	5.6	9.3	23.2	62.1
	Critical Speed (rad/s)	471	262	241	220
	Max Speed (rad/s)	1340	790	746	589
	Nominal Torque (Nm)	0.30	0.53	0.62	1.39
	Max Peak Torque (Nm)	1.03	1.72	2.01	4.43
Reducer					
	Mass (kg)	0.11	0.18	0.31	0.48
	Inertia ($\times 10^{-6}$ kg·m^2)	3.3	7.9	19.3	41.3
	Gear Ratio	100	100	100	120
	Efficiency	0.7	0.7	0.7	0.7
	Max Peak Torque (Nm)	36	70	107	217
	Max Input Speed (rad/s)	890	764	681	586
	Torsional Stiffness ($\times 10^4$ Nm/rad)	0.47	1.00	1.60	3.10
Assembly					
	Total Mass (kg)	0.69	1.11	1.55	2.90

APPENDIX C EXAMPLE MOTOR AND REDUCER CATALOGS

Table C.1 Example motor catalog with corresponding model parameters.

Motor #	**Mass** (kg)	**Inertia** ($\times 10^{-6}$ kg·m^2)	**Critical Speed** (rad/s)	**Max Speed** (rad/s)	**Nominal Torque** (Nm)	**Peak Torque** (Nm)
1	0.076	5.6	471	1344	0.30	1.03
2	0.135	9.3	262	790	0.53	1.72
3	0.162	23.2	241	746	0.62	2.01
4	0.292	32.7	131	417	1.22	3.92
5	0.356	62.1	220	589	1.39	4.43
6	0.630	108.0	115	336	2.56	8.31
7	0.712	128.6	99	307	2.87	9.20
8	0.822	148.0	84	269	3.30	10.64

Table C.2 Example reducer catalog with corresponding model parameters

Reducer #	**Mass** (kg)	**Inertia** ($\times 10^{-6}$ kg·m^2)	**Gear Ratio** (-)	**Efficiency** (-)	**Peak Torque** (Nm)	**Max Input Speed** (rad/s)	**Torsional Stiffness** ($\times 10^4$ Nm/rad)
1	0.11	3.3	50	0.7	23	890	0.34
2	0.11	3.3	80	0.7	30	890	0.47
3	0.11	3.3	100	0.7	36	890	0.47
4	0.18	7.9	50	0.7	44	764	0.81
5	0.18	7.9	80	0.7	56	764	1.00
6	0.18	7.9	100	0.7	70	764	1.00
7	0.18	7.9	120	0.7	70	764	1.00
8	0.31	19.3	50	0.7	73	681	1.30
9	0.31	19.3	80	0.7	96	681	1.60
10	0.31	19.3	100	0.7	107	681	1.60
11	0.31	19.3	120	0.7	113	681	1.60
12	0.31	19.3	160	0.7	120	681	1.60
13	0.48	41.3	50	0.7	127	586	2.50
14	0.48	41.3	80	0.7	178	586	3.10
15	0.48	41.3	100	0.7	204	586	3.10
16	0.48	41.3	120	0.7	217	586	3.10
17	0.48	41.3	160	0.7	229	586	3.10
18	0.97	169.0	50	0.7	281	503	5.20
19	0.97	169.0	80	0.7	395	503	6.70
20	0.97	169.0	100	0.7	433	503	6.70
21	0.97	169.0	120	0.7	459	503	6.70
22	0.97	169.0	160	0.7	484	503	6.70
23	1.87	450.0	50	0.7	523	419	10
24	1.87	450.0	80	0.7	675	419	13
25	1.87	450.0	100	0.7	738	419	13
26	1.87	450.0	120	0.7	802	419	13
27	1.87	450.0	160	0.7	841	419	13

APPENDIX D OPTIMIZED DESIGNS FOR INVERSE DESIGN PROBLEMS

Table D.1 Optimized design variables and objective value for Problem 1

DP or Function	**Value**
Actuator 1	XLarge/Large
Actuator 2	XLarge
Actuator 3	Medium
Actuators 4-6	Small
Dynamic Payload Capacity	20.40 kg

Table D.2 Optimized design variables and objective and constraint values for Problem 2

DP or Function	**Value**
$T_{y,3}$	-0.313 m
$T_{y,4}$	0.050 m
$T_{z,5}$	-0.363 m
$T_{y,6}$	0.050 m
$T_{z,7}$	-0.224 m
Maximum Static Payload Capacity	1.33 kg
Maximum Reach	1.00 m
GCI	0.063

Table D.3 Optimized design variables and objective and constraint values for Problem 3

DP or Function	**Value**
Actuator 1	Medium
Actuator 2	Large
Actuators 3-6	Small
Mechanical Work	37.53 J
Dynamic Payload Capacity	8.09 kg

Table D.4 Optimized design variables and objective value for Problem 4

DP or Function	**Value**
Joint 1-2 Motor	Motor 5
Joint 3-6 Motor	Motor 1
Joint 1 Reducer	Reducer 18
Joint 2 Reducer	Reducer 23
Joint 3 Reducer	Reducer 6
Joint 4 Reducer	Reducer 2
Joint 5 Reducer	Reducer 3
Joint 6 Reducer	Reducer 1
Min. Cycle Time	0.547 s

Table D.5 Optimized design variables and objective and constraint values for Problem 5

DP or Function	**Value**
$T_{z,1}$	0.000 m
$T_{y,2}$	0.065 m
$T_{z,2}$	-0.046 m
$R_{x,2}$	48.7°
$T_{y,3}$	0.009 m
$T_{z,3}$	-0.075 m
$R_{x,3}$	46.8°
$T_{y,4}$	0.181 m
$T_{z,4}$	-0.184 m
$R_{x,4}$	125.8°
$T_{y,5}$	-0.554 m
$T_{z,5}$	-0.114 m
$R_{x,5}$	19.7°
$T_{y,6}$	0.001 m
$T_{z,6}$	0.061 m
$R_{x,6}$	94.5°
$T_{z,7}$	0.000 m
Robot Mass	7.55 kg
Average Dexterity	100%

Table D.6 Optimized design variables and objective and constraint values for Problem 6

DP or Function	**Value**
Joint 1-6 Motor	Motor 1
Joint 1 Reducer	Reducer 7
Joint 2 Reducer	Reducer 12
Joint 3 Reducer	Reducer 3
Joint 4-6 Reducer	Reducer 2
$T_{z,1}$	0.051 m
$T_{z,2}$	-0.082 m
$T_{y,3}$	-0.388 m
$T_{y,4}$	0.088 m
$T_{z,5}$	-0.283 m
$T_{y,6}$	0.064 m
$T_{z,7}$	-0.078 m
Robot Mass	7.61 kg
Task 1 Reachability	100%
Task 3 Dynamic Payload Capacity	7.59 kg
GCI	0.07
Global Static Payload Capacity	1.1 kg