Master's Thesis

Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Computer Science

Energy Optimization of Robotic Cells using Machine Learning

Bc. Petr Ungar

Supervisor: doc. Ing. Přemysl Šůcha, Ph.D.
August 2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Ungar** Jméno: **Petr** Osobní číslo: **466329**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Umělá inteligence**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Optimalizace spotřeby energie robotických buněk využitím metod strojového učení

Název diplomové práce anglicky:

Energy Optimization of Robotic Cells using Machine Learning

Pokyny pro vypracování:

The energy consumption of industrial robots is an essential topic for the industry nowadays. The energy consumed by robots can be decreased by harmonizing the movements of robots [2], i.e., adjusting the timing parameters of the trajectories. Nevertheless, such optimization requires precise knowledge about robots' parameters. The hypothesis associated with this thesis is that precise knowledge is not needed and can be approximated. This thesis aims to propose a machine learning-based model for approximating the relation between the duration of the movement and its energy consumption (energy profile). The assignment defines the following tasks:
1) survey the existing literature related to the energy optimization of robotic cells,
2) using simulation analyze the energy consumption of robotic manipulators,
3) propose a machine learning model suitable for approximating the so-called energy profile,
4) compare the approximation with actual consumption (from simulation), analyze its impact on an optimization algorithm, and compare your method with existing approaches.

Seznam doporučené literatury:

[1] Michele Gadaleta, Marcello Pellicciari, Giovanni Berselli, Optimization of the energy consumption of industrial robots for automatic code generation, Robotics and Computer-Integrated Manufacturing, Volume 57, 2019, Pages 452-464.
[2] Libor Bukata, Přemysl Šůcha, Zdeněk Hanzálek, Optimizing energy consumption of robotic cells by a Branch & Bound algorithm, Computers & Operations Research, Volume 102, 2019, Pages 52-66.
[3] Mingyang Zhang, Jihong Yan, A data-driven method for optimizing the energy consumption of industrial robots, Journal of Cleaner Production, Volume 285, 2021.

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Přemysl Šůcha, Ph.D. katedra řídicí techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **15.08.2023** Termín odevzdání diplomové práce: **26.05.2023**

Platnost zadání diplomové práce: **22.09.2024**

doc. Ing. Přemysl Šůcha, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____ _____

Datum převzetí zadání Podpis studenta

Acknowledgements

First and foremost, I would like to extend my gratitude to doc. Ing. Přemysl Šůcha, Ph.D., for his invaluable guidance and supervision throughout the journey of my thesis. While the feedback at times was challenging and pushed me to my limits, it was essential in molding the final piece. I have deeply benefited from the wisdom and experience he so generously shared from his years in the academic realm. The collaboration was truly an enriching experience, and I am immensely thankful for it.

Furthermore, I owe a huge debt of gratitude to my friends and family. Their relentless support, occasional butt-kicking, and constant encouragement prevented me from losing sight of the end goal and ensured that I remained grounded and focused.

To all, thank you for being by my side and for helping me traverse this academic endeavor.

Declaration

I declare that I developed the master's thesis entitled Energy Optimization of Robotic Cells using Machine Learning independently and have listed all the used literature.

In Prague, 15. August 2023

.....
signature

Abstract

Energy efficiency in industry is an increasingly important topic, especially in the context of higher electricity costs and the drive for environmentally sustainable production. As industrial robots are a large consumer of energy, it is essential to find ways to make their operation more efficient. This thesis extends the use of the Process Simulate eRobot plugin, which currently only supports power optimization for robots made by KUKA, due to the way energy profiles are obtained. We propose a new method for obtaining these profiles using machine learning. These profiles can then be used in the eRobot plugin for energy optimization. Experimental results show that the new approach is comparable to the existing one in terms of energy savings, suggesting wider applicability to other robotic systems.

Keywords: Energy Optimization, Industrial Robots, Machine Learning

Supervisor: doc. Ing. Přemysl Šůcha, Ph.D.

Abstrakt

Energetická účinnost v průmyslu je stále důležitější téma, zejména v souvislosti s vyššími náklady na elektřinu a snahou o ekologicky udržitelnou výrobu. Protože jsou průmysloví roboti velkým spotřebitelem energie, je nezbytné najít způsoby, jak jejich provoz zefektivnit. Tato práce se zabývá rozšířením využití Process Simulate eRobot pluginu, který v současné době podporuje optimalizaci spotřeby pouze pro roboty od výrobce KUKA, a to kvůli způsobu získávání energetických profilů. Navrhujeme novou metodu získávání těchto profilů využívající strojového učení. Tyto profily mohou být následně využity v eRobot pluginu pro energetickou optimalizaci. Experimentální výsledky ukazují, že nový přístup je srovnatelný se stávajícím v míře energetické úspory, což naznačuje širší použitelnost pro jiné robotické systémy.

Klíčová slova: Energetická Optimalizace, Průmysloví Roboti, Strojové Učení

Překlad názvu: Optimalizace spotřeby energie robotických buněk využitím metod strojového učení

Contents

Acronyms **1**

1 Introduction **3**
1.1 Motivation 3
1.2 Related Work 4
1.2.1 Indirect EC Optimization 5
1.2.2 Direct EC Optimization 5
1.2.3 Use of Machine Learning 6
1.3 Contribution 6
1.4 Outline 7

2 Simulation Setup **9**
2.1 Siemens Process Simulate 9
2.1.1 Operation Tree 9
2.1.2 Used Extensions 10

3 Problem Statement **13**
3.1 Formal Problem Definition 14
3.2 Energy Profile 15
3.2.1 Linear Approximation 16
3.3 Idle Energy Consumption 17
3.4 MILP Model 17

4 Energy Profile Analysis **19**
4.1 Data Gathering 19
4.2 Feature Extraction and Engineering 21
4.3 Key Findings 23

5 Energy Profile and Idle Consumption Approximation **29**
5.1 Estimator Design 29
5.2 Estimator Implementation 31

6 Experimental Results **35**
6.1 Experimental Setup 35
6.2 Results 36
6.3 Summary 40

7 Conclusion **43**
7.1 Future Work 44

Bibliography **45**

Figures

1.1 Energy consumption development [1] . . . 3
1.2 Example of robotic cell with 2 IRs [2] . . . 4

2.1 Example of OT as represented in PS [2] . . . 10
2.2 KRC outputs . . . 11
2.3 Optimization plugin . . . 12

3.1 Example of optimization graph [2] 13
3.2 Examples of energy profiles 15
3.3 Approximated EP . . . 16

4.1 Simulation Setup . . . 19
4.2 Example of energy profile . . . 23
4.3 Relationship between steepness and measured energy saving . . . 24
4.4 Two EPs with same steepness . . 25
4.5 Correlation matrix for static features . . . 26
4.6 Correlation matrix for dynamic features of $j_1 - j_3$ . . . 26
4.7 Correlation matrix for dynamic features of $j_4 - j_6$ . . . 27

5.1 Predicted energy saving compared to measured energy saving . . . 31
5.2 Predicted optimal duration compared to measured optimal duration . . . 32
5.3 Predicted energy penalty at min speed compared to measured energy penalty at min speed . . . 33
5.4 Predicted profiles and original measurements offset to overlap . . . 34

6.1 Linearly approximated EP with 4 segments used in KRC optimization [3] . . . 37
6.2 Optimized OTs . . . 38
6.3 Random experiment – CT $\approx 2 \cdot d_{min}$ . . . 39

Tables

5.1 Relative and Absolute errors of predictors . . . 32

6.1 Average relative energy saving . . 36
6.2 Simulation results – 110-133 % d_{min} . . . 41
6.3 Simulation results – 150-200 % d_{min} . . . 42

Acronyms

CT cycle time. viii, 4, 5, 7, 17, 18, 29, 33, 35–40

EC energy consumption. 4–6, 10, 11, 13, 15–17, 19, 21–25, 29–33, 35, 36, 38, 40–42

EP energy profile. viii, 5–7, 10, 11, 13–17, 19, 20, 23–25, 29, 30, 32, 33, 35–37, 39, 40

eRobot eRobot Project – Czech Technical University. vi, 6, 7, 9, 11, 13, 16, 29, 30, 43

IR industrial robot. viii, 3–6, 9, 11, 14, 17, 19, 37

KRC Kuka Robot Controller. viii, 6, 7, 9, 11, 13, 14, 29, 31, 35–43

MILP Mixed Integer Linear Programming. 13, 14, 16, 17, 29, 37, 40

OT operation tree. viii, 9–11, 14, 19, 20, 24, 35–40

PS Siemens Process Simulate. viii, 6, 7, 9–11, 14

PtP point-to-point. 10, 14, 21, 30

RRS Realistic Robotic Simulation. 11

TCPF Tool Center Point Frame. 10, 17, 21, 22, 31

Chapter 1

Introduction

According to the International Energy Agency (IEA) [1], as seen in Figure 1.1, the overall energy consumption witnessed an increase of approximately 80 % between 2000 and 2019, with the industrial sector alone experiencing nearly a doubling of consumption. Despite the slowdown caused by the COVID-19 pandemic in 2020, the energy consumption continues to surge [4]. In addition, in recent times, electricity prices have seen a significant rise due to various factors such as the global pandemics and geopolitical situation. Although the impact is felt worldwide, Europe is the most affected region. The price of electricity in Europe has increased almost sixfold [5]. While analysts anticipate a decrease in prices, they still expect them to remain two to three times higher than the 2020 levels. In light of these circumstances, coupled with mounting pressure from regulatory bodies to reduce emissions, energy optimization is poised to become increasingly vital from both financial (electricity savings) and regulatory (emission reduction) perspectives.

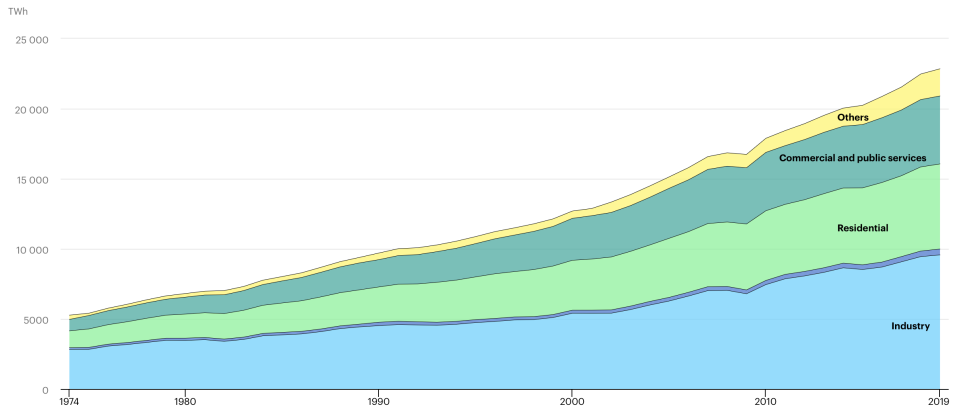


Figure 1.1: Energy consumption development [1]

1.1 Motivation

Industrial robots (IRs), the workhorses of modern manufacturing, are significant consumers of energy [6]. In a typical scenario, these machines are

programmed to operate at their maximum speed limit, only to then wait idly until the next task is assigned. This approach, while straightforward, is far from efficient. Given the nature of production lines and their inherent bottlenecks, many robotic cells do not need to operate at full speed to meet the production line's cycle time (CT).

As we will explore in this thesis, adjusting the parameters of the robot's movement, specifically the speed limit, can have a profound impact on energy consumption (EC). By utilizing more time to execute tasks and reducing idle waiting periods, we can achieve a more energy-efficient operation. However, finding the optimal solution is not a trivial task, especially when dealing with multiple robots within a single cell (such as in Figure 1.2).

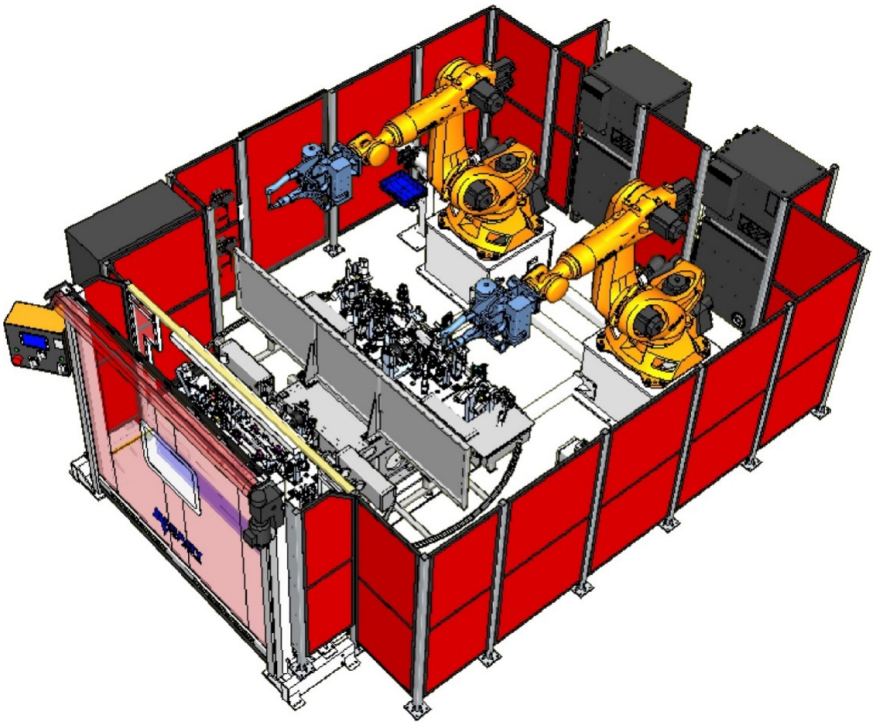


Figure 1.2: Example of robotic cell with 2 IRs [2]

In this thesis, we explore the application of machine learning in enhancing energy efficiency in industrial robotics. We propose a predictive model that estimates energy consumption profiles for robotic movements, bypassing vendor-specific energy optimization solutions, thus promoting wider applicability. The study signifies an important step towards universal, energy efficient practices in the field of industrial robotics, contributing to the broader goal of achieving a more sustainable future.

1.2 Related Work

In this section, we present a review of current strategies and the variety of methods aimed at optimizing the EC of IRs. While some of these techniques

are directly centered on energy optimization (Direct EC Optimization), others achieve energy efficiency as a beneficial side effect or byproduct (Indirect EC Optimization).

1.2.1 Indirect EC Optimization

Historically, the energy consumption of IRs has not been a primary concern, as the majority of the focus has been placed on optimizing the production CT. However, it is important to note that reducing CT can positively affect EC in certain instances. There exist various approaches to time optimization, such as determining the optimal sequence of operations [7], or planning the optimal trajectory [8]. These techniques can help reduce the operational duration of IRs, which in turn, can lead to lower energy usage. Specifically, planning the optimal trajectory can minimize unnecessary movements and optimize motor usage, reducing energy waste. Similarly, an optimally sequenced operation can reduce idle time, which also contributes to energy conservation. In a comprehensive study, Stuhlenmiller et al. [9] explored the correlation between CT, EC, and general resource efficiency, discussing the benefits and challenges associated with reducing cycle time, such as potential cost reduction per work-piece versus an increased probability of requiring spare parts.

1.2.2 Direct EC Optimization

Given the evolving dynamics of energy markets and the increasing pressures from regulatory entities advocating for sustainable manufacturing, there is an imminent need for focused research into energy optimization. Several publications have already begun to prioritize EC optimization [10], [11]. The research by Michele Gadaleta et al. [12], [13] introduces several compelling concepts, most notably the energy profile (EP) (the relationship between consumption and duration of the motion), which is extensively employed in this thesis. In [12], EC optimization methods are classified into two categories:

- *Eco Design methods* are applied before the actual production line is constructed. These may encompass the optimal selection of IRs and their ideal placement. The main limitation with these methods is that they are not applicable to existing production lines. As energy efficiency was not a major concern until recently, these methods are not useful for optimizing the EC of already established, energetically sub-optimal production lines.
- *Eco Programming methods*, in contrast, can be employed on existing production lines. These strategies commonly address the issue of optimal scheduling and trajectory planning. The diverse approaches to Eco programming will be further examined.

The first strategy emphasizes optimizing the consumption of each IR independently. The typical focus within this approach is energetically optimal trajectory planning. For instance, Liu et al. [14] employed numerical methods

to identify energy optimal paths, reporting a substantial energy reduction of up to 15.3 %. Additional research on energy optimal trajectory planning can be found in [15], [16], demonstrating the effectiveness of this strategy in reducing the energy consumption of individual robots.

An alternative strategy prioritizes optimizing the robotic cell as a whole rather than individual robots. The solution proposed by Bukata et al. [17] considers each robot's position and movement, along with their respective EP, to optimize the entire robotic cell. They report that practical application of this approach in Škoda Auto led to 20 % energy savings in robot movements.

The study carried out by Paryanto [18] validated the efficacy of using simulation tools for energy optimization. They established the credibility of the simulated data by cross-verifying it with data obtained from real-life experiments, confirming that the simulation tools can generate reliable information for energy optimization tasks. Furthermore, it was also observed that the EC is influenced by operational parameters, primarily speed limits. This assertion is supported by research of Bukata, Šůcha, Hanzálek [17] and the work of Petr [3]. Additionally, the ongoing research led by Šůcha resulted in a Siemens Process Simulate (PS) plugin for optimizing digital twins, initially as part of the eRobot Project – Czech Technical University (eRobot) and later extended by Petr [3]. The developed solution effectively utilized the concept of EPs, but the plugin heavily relies on extensive simulations that often require significantly more computational resources than the optimization process itself. The most prominent limitation of this approach, however, is the plugin's dependency on the Kuka Robot Controller (KRC) plugin to obtain the EPs. Consequently, the applicability of this plugin is limited to KUKA robotic systems, restricting its broader usage in diverse robotic ecosystem.

1.2.3 Use of Machine Learning

The application of machine learning techniques to this problem domain has shown considerable potential. For instance, Zhang and Jihong's study [19] utilized shallow neural networks to predict the EC of Epson C4 6-DOF IR. Remarkably, they achieved an average deviation in predicted EC of 4.59 %, even with a relatively small dataset of fewer than 800 samples. However, their research has its share of limitations. The study lacks a thorough examination of the variables influencing energy consumption, potentially undermining the robustness of their findings and their applicability to real-world scenarios. Therefore, the focus of our research is to enhance the understanding of these factors and improve the robustness and generalizability of the predictive models for EC.

1.3 Contribution

This thesis builds upon the foundation laid by the eRobot team, the work of Bukata et al. [17], and Petr [3]. Like these previous works, it employs the same model for energy optimization, with all optimization and simulation processes

performed in PS. However, this thesis introduces significant simplifications and generalizations to the process. Our comprehensive examination of EPs has allowed us to bypass the use of the KRC, previously required for obtaining the EPs. Instead, we developed a machine learning models that allow us to predict the EPs.

The simulation results indicate that our method slightly outperforms the KRC-based method for small increases in CT and slightly under-performs for larger increases in CT. Importantly, our approach also decreases the number of initial PS simulations needed to construct the EP from five to two, improving overall optimization duration. This contribution represents a major advance in the field of energy optimization for robotic cells, underscoring the value of machine learning in this domain.

We emphasize that the simulations conducted for this research were focused on a basic robotic cell featuring a specific model of a KUKA robot. To extend the applicability of our methodology to other robot models, additional simulations or measurements would be necessary to provide data for training. This is particularly important given that energy consumption can differ significantly between robot models.

1.4 Outline

This thesis is organized into six chapters. Chapter 1 offers an overview of related work and provides a contextual understanding of the current economic landscape to elucidate the motivation driving this study. Subsequently, Chapter 2, describes the simulation setup used in this research and outlines the associated plugins and third-party tools required. In Chapter 3, we formally define our problem and place it within the context of PS and the existing eRobot plugin. Chapter 4 and Chapter 5 delve into the problem at hand, discussing the different strategies considered, the reasoning behind our chosen method, and the process of data gathering, manipulation, and analysis as well as implementation of profile prediction. Finally, in Chapter 6 and Chapter 7, we present our results, draw conclusions, and discuss potential enhancements for future research.

Chapter 2

Simulation Setup

In the preceding chapter, we stated that the experimental setup for this thesis aligns with the previous work conducted by the eRobot team. In this chapter, we will further detail this setup, introducing the concept of a digital twin and the software used to operate it, along with the third-party components that enhance its functionality.

According to Grieves [20], a digital twin is a virtual representation of a physical system, such as a 3D model of a robotic cell. However, a digital twin is not limited to 3D models and can include various digital forms that mirror physical systems. The purpose of a digital twin is to emulate the behavior and performance of its physical counterpart, serving as an invaluable tool for programming, simulating, monitoring, and optimizing system operations.

There are numerous robotic simulation software options available, often provided by leading robotic vendors. For example, KUKA offers the KUKA.Sim software, while ABB provides the RobotStudio software. However, this thesis uses exclusively Siemens Process Simulate (PS), that is why we will describe how this software represents operations and how we interact with it.

2.1 Siemens Process Simulate

PS is a software developed by Siemens. The primary aim of PS is to aid manufacturers in creating virtual models of IRs and other physical systems, which are then used for designing, simulating, and optimizing production processes. Let's first discuss how a manufacturing process is represented in PS, followed by description of the two crucial plugins, used to extend the basic functionality to accommodate our requirements, the KRC and eRobot plugins.

2.1.1 Operation Tree

An operation tree (OT) in PS (Figure 2.1) is a hierarchical tree structure that outlines the steps involved in a manufacturing process. In a general sense, an OT consists of two types of operations:

- **Compound Operations (non-leaf nodes)**: These are groups of operations that keep related operations together, simplifying the overall

process. Though a compound operation itself does not represent any actual movement, it contains a mix of simpler (atomic) operations or other compound operations to abstract more complex movements. One example could be a long transition, where several *via* points are added to control the route. A *via* point in the context of a movement or transition is an arbitrarily chosen intermediate position or waypoint used to shape the path or trajectory. Though these points guide the route, they aren't directly linked to the specifics of the underlying manufacturing process. We would then nest all these operations under one compound operation to hide the complexity. The root of a OT is always a compound operation.

- **Atomic Operations (leaf nodes)**: These operations represent the actual physical movements in a manufacturing process. They sit at the bottom of an OT and act as building blocks for the more complex compound operations. Examples include a linear movement, which moves in a straight line, or a point-to-point (PtP) movement, which moves a robot's Tool Center Point Frame (TCPF) from one location to another. This study focuses solely on PtP movements, as they are the movements that we can typically adjust because they are not a specific part of the manufacturing process. Moreover, PtP movements allow us to decide how far from given coordinates the TCPF needs to move. This setting is known as the *zone* and is usually used for the aforementioned *via* points. Here, we want to change the route but do not need to reach the exact spot. This can be helpful as it allows the controller to find the best route between the start and end of a trajectory while following the general direction of the predefined route (via points). We generally distinguish between *fine* (exact location) and *coarse* (within some radius) movement, which becomes important later when we define the optimization problem and derive the respective EPs.

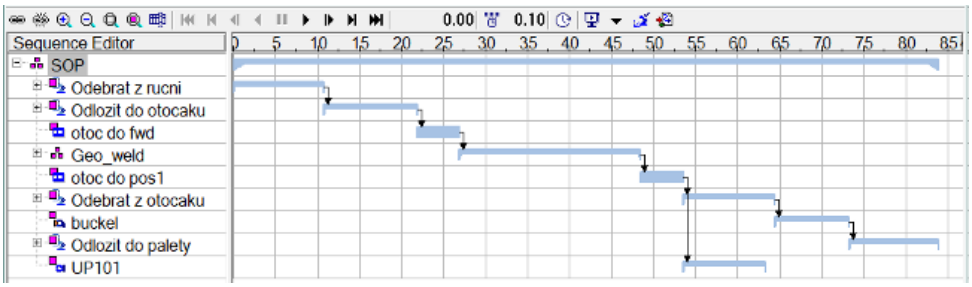


Figure 2.1: Example of OT as represented in PS [2]

2.1.2 Used Extensions

The default configuration of PS lacks the features needed for energy optimization. In particular, it does not have the capability to estimate EC or accurately plan trajectories, thus it cannot offer precise estimates of duration

for a given movement. To address these limitations, we have incorporated the KRC plugin developed by KUKA, which offers Realistic Robotic Simulation (RRS). An RRS software aims to accurately mirror the behavior of real-world robotic systems in a simulation. It incorporates the physical attributes of robots, such as weight, size and reach, to enhance the realism of the simulation. The KRC tool is provided as a black box and provides more accurate control over KUKA robots compared to the default PS controller. In our work, the primary use of KRC is as a source of energy usage estimates, but it also provides information on joint velocities and accelerations (as demonstrated in Figure 2.2). Its use, however, also represents a significant drawback to the current solution (eRobot plugin) as its compatibility is restricted to KUKA robots, limiting its broader applicability to IRs from other vendors.

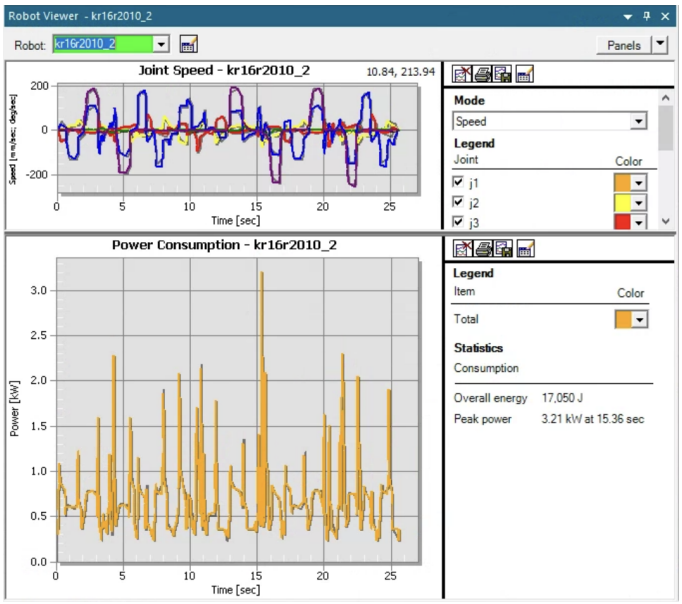


Figure 2.2: KRC outputs

The second plugin is the eRobot plugin (as seen in Figure 2.3). It implements the actual optimization as originally developed in [3], [17]. This plugin works directly with PS, first obtaining the OT that defines the process. It then instructs the simulation to perform several runs at various speed limits, gathering consumption data from KRC, which are then used to derive the EPs. With these profiles in hand, it conducts the optimization, subsequently inserting the speed limits and wait times back into the OT. The updated OT can then be run with optimal EC without needing further modification. The aim of our work is to modify a portion of this plugin's logic to establish a new solution that could operate independently of the KRC. We also use a customized version of this plugin to generate the dataset for this thesis.

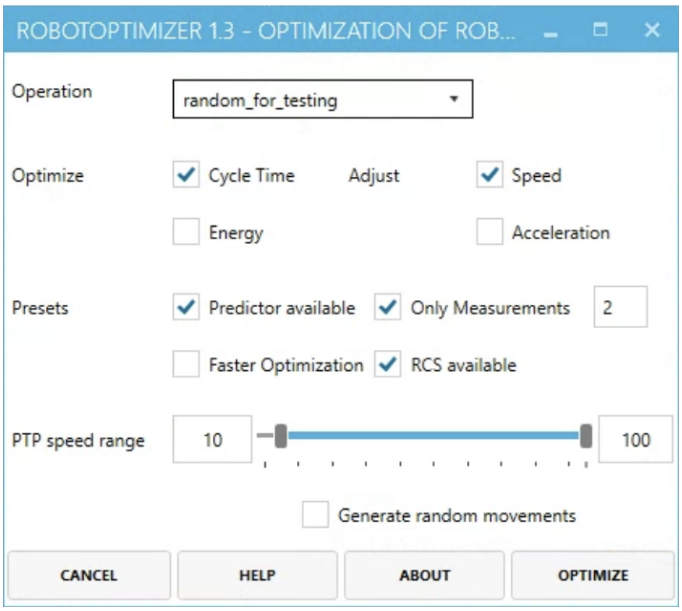


Figure 2.3: Optimization plugin

Chapter 3

Problem Statement

This chapter begins with a formal definition of the problem, followed by description of polynomial EP and its transformation into a linear approximation directly used in the optimization. The chapter concludes with a discussion of the key elements of the original Mixed Integer Linear Programming (MILP) formulation.

This thesis's main objective is to replace the existing procedure for deriving EPs in the eRobot plugin. Currently, the process involves measuring EC at various speed limits (utilizing KRC), followed by interpolating and linearly approximating these measurements. Instead, the aim here is to predict EPs directly, offering several benefits, such as eliminating the current vendor lock-in on KUKA robots and potentially accelerating the process by reducing number of initial simulations to obtain a profile.

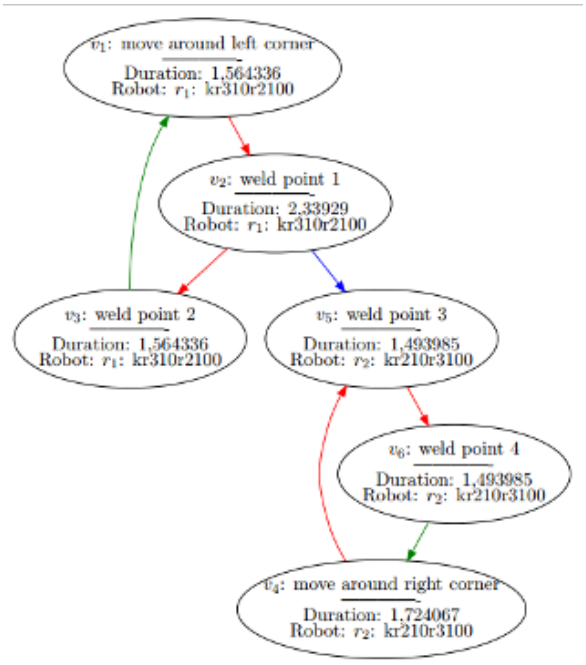


Figure 3.1: Example of optimization graph [2]

3.1 Formal Problem Definition

The optimization problem is identical to the problem described by Bukata et al. [17] and later refined by Petr [3]. The focus of this thesis is solely on obtaining EPs without the necessity for KRC.

The MILP problem definition is intimately linked to the OT concept, detailed in Subsection 2.1.1. We define a set of robots $R = \{r_1, ...r_{|R|}\}$, a set of vertices $V = \{v_1, ...v_N\}$, edges $E = \{e_1, ...e_M\}$, and a graph $G = (V, \overline{E}, CR)$ (Figure 3.1).

Each vertex $v \in V$ can be either atomic or a collection of atomic operations. The decision depends on the *zone* attribute of the operations (Subsection 2.1.1). If the zone is *fine*, indicating that the robot must reach the precise location, the operation can independently form a vertex. Conversely, if the zone is *coarse*, such operations must be succeeded by a *fine* point. Therefore, a vertex can be a *fine* zone operation or a set of $\{coarse_1, ..., coarse_n, fine\}$ zone operations. The later is common for longer movements, these are typically partitioned into $FROM$ and TO locations with several *via* points in between, ensuring a collision-free path, non-singular configuration, or enforcing other manufacturing requirements. To describe a vertex's execution state, we differentiate between three vertex phases: prior-execution N (before the vertex-induced movement starts), execution/movement M (during vertex execution), and post-execution W (after completion of vertex execution). With these phases, we also introduce variables $dp_v, dw_v, \underline{dm}_v, \overline{dm}_v$, respectively denoting prior-execution wait, post-execution wait and the minimum vertex duration and maximum vertex duration. The concept of prior/post waits was introduced for collision resolution but can also encode specific underlying process requirements, such as glue hardening or screwing. During optimization, these waits are also employed if it's determined that pausing is more energy-efficient than further reducing movement speed.

The set $\overline{E} = E \bigcup L$ signifies the order of vertices, where E and L are sets of directed edges/tuples between two vertices $e_i = (u, v)$. The distinction between sets E and L lies in that for all $e \in E : R(u) = R(v)$ and for all $l \in L : R(u) \neq R(v)$, meaning E represents the vertex order for one robot, whereas L denotes the vertex order of different robots.

To simplify the problem, this thesis opts to allow only *fine* zone settings for all PtP movements, meaning that each optimization vertex will contain only a single operation. The study also only focuses on a single type of robot, specifically the KUKA KR16R2010, due to the time-intensive nature of obtaining and configuring digital twins of different robot types.

Lastly, the collision resolution set, represented as CR, consists of quadruples $\{v_u, v_v, op_u, op_v\}$, which identify potential points of collision between two vertices, v_u and v_v, during their corresponding operation phases, op_u and op_v. This set functions as an internal tool for resolving any collisions arising due to alterations in speed limits. Given that we solely consider one IR, this set is always empty.

In a nutshell, a PS OT (a graph) is transformed into optimization graph

used throughout the optimization. The primary difference between the two is that in the optimization graph, multiple atomic (leaf) nodes can be grouped into one, based on operation zone. Furthermore, in the optimization graph, additional variables are introduced to denote waiting before or after executing a given node, along with the minimum and maximum duration for the movement. These nodes are then interconnected with edges and links, representing the execution order.

3.2 Energy Profile

Each optimization vertex $v \in V$ is associated with single EP. This profile describes the relationship between the duration of a given operation or set of operations (vertex) and the corresponding EC. The vertex duration can be adjusted by altering the speed limit. An EP is typically a convex function that can be approximated by a polynomial of the form:

$$E(d) = \sum_{i=-2}^{1} a_i \cdot d^i. \tag{3.1}$$

The order of the polynomial can be modified, however the provided polynomial is sufficiently accurate to describe majority of EPs. The profile is found by conducting several measurements at different speed limits and then interpolating these measurements using the specified polynomial. Examples of what an EP may look like are depicted in Figure 3.2, where individual measurements are denoted by red dots and the final profile is represented by the blue line. It's also evident from the figure that the profiles can vary greatly.

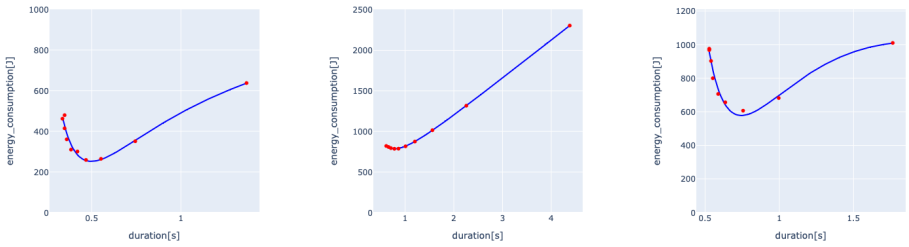


Figure 3.2: Examples of energy profiles

While the continuous polynomial function is presented and illustrated in several figures, it is important to note that this thesis does not actively utilize the continuous approximation. The inclusion of the polynomial EP definition is primarily to provide a clear context with previous work. The primary reason why we can avoid using the polynomial interpolation is the abundance of measurements we conducted. In theory, it is possible to obtain the entire EP by measuring EC for a sufficiently large number of (infinitely many) speed limits. The initial introduction of the polynomial EP [17] was mainly

motivated by time-saving considerations, as the initial simulations can be computationally expensive. However, due to the ability to perform a greater number of measurements without being overly concerned about the duration of data gathering, we were able to forgo the use of polynomial EPs.

3.2.1 Linear Approximation

While the EP is generally modelled by a polynomial function, the computational tool of choice, MILP solver, requires linear constraints. Hence, there is a need to convert the polynomial into an approximated linear form for use within the solver. To approximate the polynomial with a linear function, the plugin utilizes piece-wise linear approximation. The polynomial is approximated by a series of line segments. Each line segment approximates the polynomial over a specific interval (as seen in Figure 3.3). As the number of line segments increases, the approximation becomes more precise. However, this comes at the expense of increased computational demands, thereby creating a trade-off between approximation accuracy and computational complexity. The number of segments utilized is adjustable and is one of the input parameters for the eRobot plugin. We determine the approximated EC by reading the y-value of the segment approximating interval to which the duration (x-value) falls.

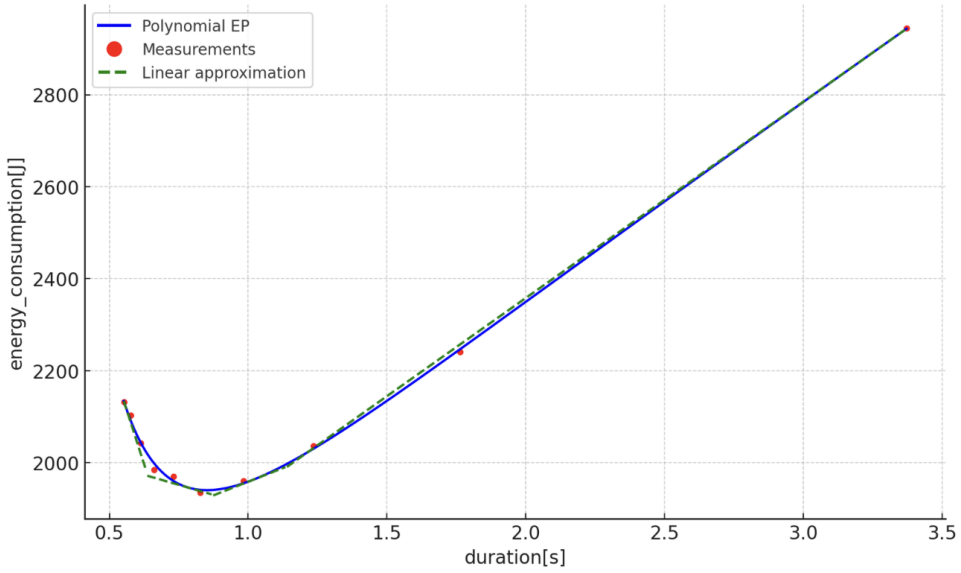


Figure 3.3: Approximated EP

3.3 Idle Energy Consumption

It is crucial to comprehend that during a cycle, the robot's EC can be impacted not only by the motion. Substantial time may also be spent in idle states, whether to satisfy CT requirements or due to the manufacturing process demands like drilling, welding, etc. During these periods, the robot still uses a considerable amount of energy.

Just like with EP, the idle power consumption is also assigned to each optimization vertex, reflecting its variability based on the robot's idle location. For instance, EC appears to be higher when the TCPF extends further from the base compared to when it's closer. In practical scenarios, this consumption may not be a constant, as many IRs offer power-saving modes (such as mechanical brakes) that kick in after a certain idle period, but for our purposes we consider it to be constant.

3.4 MILP Model

Finally, we arrive at the definition of the MILP problem. As mentioned earlier, this formulation aligns with the problem descriptions presented in prior research [3], [17]. Even though it is not the central focus of this thesis, it heavily influences many decisions and assumptions that underpin the entire work. Therefore, while we will not be presenting the full model, we will outline and describe its key components for better understanding.

$$\min_{\mathbf{dn},\mathbf{dm},\mathbf{dw},\mathbf{dc}} \sum_{v \in V} P_{N_v} dn_v + P_{W_v} \left(dw_v + dc_v\right) + E_v \tag{3.2}$$

s.t.

$$\text{offset}_{tv} \leq E_v + \text{slope}_{tv} dm_v, \quad \forall v \in V, \forall t \in T \tag{3.3}$$

$$s_u + dm_u + dw_u + dc_u = s_v - dn_v + h_{uv}\omega, \quad \forall\,(u,v) \in E \tag{3.4}$$

$$\underline{dm}_v \leq dm_v \leq \overline{dm}_v, \quad \forall v \in V \tag{3.5}$$

$$E_v, \underline{dm}_v, \overline{dm}_v, \omega, \text{slope}_{tv}, \text{offset}_{tv}, h_{uv}, P_{N_v}, P_{W_v} \in \text{const.}$$

The objective is to minimize the total EC. As seen in Equation 3.2, the EC is computed by summing up the total energy consumed during the movement of all vertices (E_v) and the total energy consumed during idle periods – prior wait and post wait (P_{N_v} and P_{W_v} are the waiting power consumption constants). Equation 3.3 is crucial for this thesis, as this is the condition enforcing following the EP, which is approximated using linear functions (as described in Subsection 3.2.1). Each linear segment $t \in T$ is defined by its offset and slope. Equation 3.4 constraint ensures the order of operations among vertices and robots. It states that vertex v_v can only start its operation after vertex v_u has completed its movement and done its post-waiting (dw_u).

If the edge (v_u, v_v) starts a new cycle, then h_{uv} equals 1 and the CT ω is added to the start time of vertex v_v, this enforces the CT is met. Lastly, the Equation 3.5 implies that all vertices $v \in V$ have a lower bound $\underline{dm}_v$ and an upper bound $\overline{dm}_v$ on their duration dm_v.

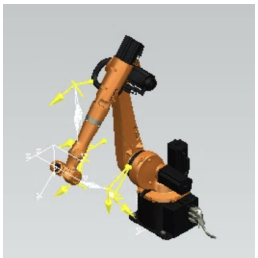
Chapter 4

Energy Profile Analysis

As mentioned in Section 3.2, an EP is a function describing the relationship between movement duration and its EC. These profiles can vary a lot and one of the main tasks in this thesis was the analysis of what influences the shape and in turn the consumption of IRs. This chapter delves into the analysis of the EPs. We will describe the collection of raw data, the process of analyzing and interpreting this data to gain understanding of the robot's EC patterns. The findings shared in this chapter are important for later predicting the EPs, which will be described in the following chapter.

4.1 Data Gathering

In an ideal scenario, we would utilize an ample number of 'real-world' simulations, mimicking actual production lines, for analyzing robotic movements. However, due to practical constraints, this approach is not viable. As a workaround, we generated our own simulations (OTs), which presented its own set of challenges. As previously mentioned in Section 3.1, all simulations were performed on single robot, specifically KUKA KR16R2010 (Figure 4.1a).



Paths & Locations	Config	Acc	Tool Nr	Base Nr	Motion	Speed	Zone
random_for_testing							
Dummy							
HOME		100 %		0 - WorldF...	PTP	100 %	FINE
via0		100 %		0 - WorldF...	PTP	100 %	FINE
via1		100 %		0 - WorldF...	PTP	100 %	FINE
via2		100 %		0 - WorldF...	PTP	100 %	FINE
via3		100 %		0 - WorldF...	PTP	100 %	FINE
via4		100 %		0 - WorldF...	PTP	100 %	FINE
via3		100 %		0 - WorldF...	PTP	100 %	FINE
via2		100 %		0 - WorldF...	PTP	100 %	FINE
via1		100 %		0 - WorldF...	PTP	100 %	FINE
via0		100 %		0 - WorldF...	PTP	100 %	FINE
HOME		100 %		0 - WorldF...	PTP	100 %	FINE

(a) : KR16R2010 **(b) :** Example of random OT

Figure 4.1: Simulation Setup

When randomly generating joint configurations, many invalid configurations are produced since valid configurations for individual joints may not result in a valid configuration when combined. To counter this, we implemented a *budget* to constrain the complexity of operations (the sum of individual joint differences). The algorithm we developed for generating random OTs offers

several advantages. Firstly, it provides control over movement complexity, ensuring a balanced dataset containing both simple and complex movements. Secondly, it significantly reduces the number of invalid configurations, despite not entirely eliminating them.

Using this algorithm, we produced several hundred random OTs, such as in Figure 4.1b, each starting from the same origin and containing five randomly generated points[1]. The final OT visits all these points in sequence up to the fifth one, then retraces the points in reverse order. The reverse traversal was implemented to enable accurate comparisons of two profiles following the same trajectory but in opposite directions. After this, we performed numerous simulations with 10 different speed limits, for every generated OT, resulting in approximately 6000 distinct EPs that we subsequently analyzed. All these simulations ran within a speed limit range of 10 % minimum and 100 % maximum.

Algorithm 1: Generating random operation tree

Input : int n, float B, bool useWholeBudget
Output : Randomly generated operation tree
// n is the number of points we wish to generate
// B is the maximum sum of changes of all joints
$P \leftarrow \emptyset$;
while $|P| < n$ **do**
  Randomly distribute the budget B between all joints;
  foreach *joint* **do**
    if *useWholeBudget* **then**
      if *flipCoin() = heads* **then**
        $joint \leftarrow joint + budget$;
      else
        $joint \leftarrow joint - budget$;
    else
      $joint \leftarrow random(0, 2 \cdot budget) + joint - budget$;
    $joint \leftarrow \min(joint, \text{upper joint limit})$; // Limit the value to the upper joint limit
    $joint \leftarrow \max(joint, \text{lower joint limit})$; // Limit the value to the lower joint limit
  Add the newly generated point to set P;
return P;

[1] all these points are using *fine* zone settings

4.2 Feature Extraction and Engineering

Each simulation generates a multitude of metrics, such as the duration and EC of each operation for given speed limit, as well as initial and final TCPF location and joint configurations, alterations in angular speed, acceleration, and more. Most of these features are not later used or analyzed directly but are modified and encoded into different features.

- **Speed Limit [%]:** This represents the speed limit set for the operation, expressed as a percentage of the maximum possible speed.
- **Duration [s]:** This feature captures the duration of each Point-To-Point (PTP) movement. The duration varies based on the speed limit and the complexity of the movement.
- **Energy Consumption [J]:** The energy consumption for each PTP movement is recorded. This together with duration are used for creating the energy profiles.
- **Cartesian and Joint Coordinates:** These are the Cartesian X, Y, Z and rotation rX, rY, rZ coordinates of the end-effector (TCPF), and the joint angles θ_{1-6} of the robot at each location. They provide a detailed configuration of the robot at each point.
- **Joint Dynamics:** For each joint (j_{1-6}) and each speed limit, the joint speed, acceleration, and jerk (rate of change of acceleration) are recorded at every simulation step. These parameters represent the dynamic behavior of the robot, and with a high simulation frequency, we get hundreds of values per second.

The raw data gathered during the simulations allowed for the construction of a diverse set of features capturing both the static and dynamic behaviour of the robot. For each PtP movement (optimization vertex) we used the measured features to generate new ones. The following features are the one selected as most influential and useful for describing the movements characteristics.

- **Measured Optimal Speed Limit [%]:** This feature denotes the measured speed limit that results in the smallest EC. It provides insights into the trade-off between speed and energy efficiency in operations.
- **Measured Optimal Duration [s]:** This is the duration of operation at the measured optimal speed limit.
- **Measured Energy Saving [J]:** This feature represents the actual energy saving achieved by operating at the measured optimal speed limit (compared to moving at 100 %).

- **Measured Energy Penalty at Min Speed [J]:** This feature measures the additional (compared to optimal EC) energy costs incurred by operating at the minimum speed limit (10 %) .
- **Duration Slowest [s] and Duration Fastest [s]:** These features capture the operation time at the slowest and fastest speed limits, respectively.
- **Theta Sum:** This represents the sum of all changes in joint configurations (θ_i), giving an overall measure of movement complexity.
- **Rotation Angle:** This is the angle of rotation of the robot's end-effector (TCPF).
- **Euclidean Distance, Vertical Distance and Horizontal Distance:** These features measure the spatial distances travelled by the end-effector during the operation.
- **Joint Speed Difference [rad/s]:** This feature captures the difference in angular speed of a given joint between the angular speed at maximum speed (100%) and minimum speed movement (10 %).
- **Joint Acceleration Difference [rad/s^2]:** This feature measures the difference in maximum acceleration of a given joint between the acceleration at maximum speed (100%) and minimum speed movement (10 %).
- **Joint Deceleration Difference [rad/s^2]:** This feature records the difference in maximum deceleration (negative acceleration) of a given joint between the deceleration at maximum speed (100 %) and minimum speed movement (10 %).
- **Joint Jerk Acceleration Difference [rad/s^3]:** This feature captures the change in the maximal jerk (rate of change of acceleration) of a given joint between the jerk at maximum speed (100 %) and minimum speed movement (10 %).
- **Joint Jerk Deceleration Difference [rad/s^3]:** This measures the change in maximal jerk of a given joint between the jerk during deceleration at maximum speed (100 %) and minimum speed movement (10 %).
- **Joint Angle Differences:** These features correspond to the absolute value of differences in each joint angle θ_i - by how much the joint configuration changed between initial and final configuration.

4.3 Key Findings

This section showcases the significant insights derived from data analysis process. The primary goal was to better understand the EPs and to identify the most influential features impacting EC.

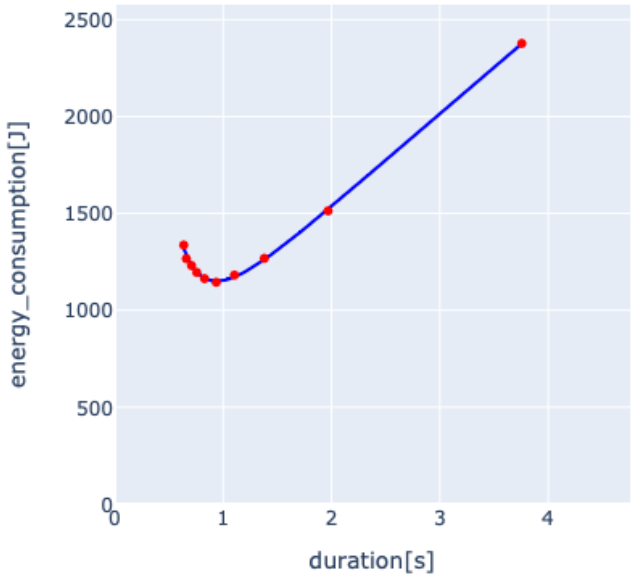


Figure 4.2: Example of energy profile

To begin, we need to define what a typical EP looks like. We will utilize Figure 4.2 for this purpose. As discussed in Section 4.1, all simulations were conducted with 10 different speed limits (represented by red dots), ranging from a maximum speed limit of 100 % to a minimum of 10 %. These limits correspond to the farthest left and right extreme values and the associated duration is the $\underline{dm}_v$ and $\overline{dm}_v$ (Equation 3.5) of a vertex, respectively. Subsequent interpolation of these points, as detailed in Section 3.2, yielded the EP (represented by the blue line). The profile shows that moving at maximum speed takes about 0.63 s and consumes approximately 1,340 J. It then quickly arrives at the optimal duration of around 0.94 s, where EPs is minimized to roughly 1,144 J. Thereafter, consumption begins to increase to nearly 2,800 J at around 3.76 s when the robot operates at its minimum speed. Notably, the EPs can be divided into three distinct segments:

- **Decreasing Consumption** segment – it spans from the shortest duration to about one measurement before the optimal duration. It is typically the steepest part of the profile and is of great interest, as it is in this section that energy savings can be achieved by lowering speed.

- **Near-Constant Consumption** segment – despite the name suggesting a stable consumption rate, this segment surrounds the optimal duration, where the profile tends to 'level out' before gradually transitioning into the final segment.
- **Increasing Consumption** segment – commencing roughly one measurement point after the optimal duration, this part of the profile shows that further deceleration of the movement leads to increased ECs. This section can be quite accurately approximated by a linear function across its entire length.

While all the three segments are found in most of the profiles, the proportionality of these segments can vary a lot, this can clearly be seen in Figure 3.2.

Now that we have identified the characteristics of a typical EP, our next step is to establish a method for comparing them. We aim to distinguish between good and bad candidates for optimization based on their 'suitability'. In this context, suitability refers to the preference or priority we assign to specific EP within a given OT when greedily decreasing speed limits. It determines the order in which we would slow down one operation over slowing down another one, in order to conserve the most energy.

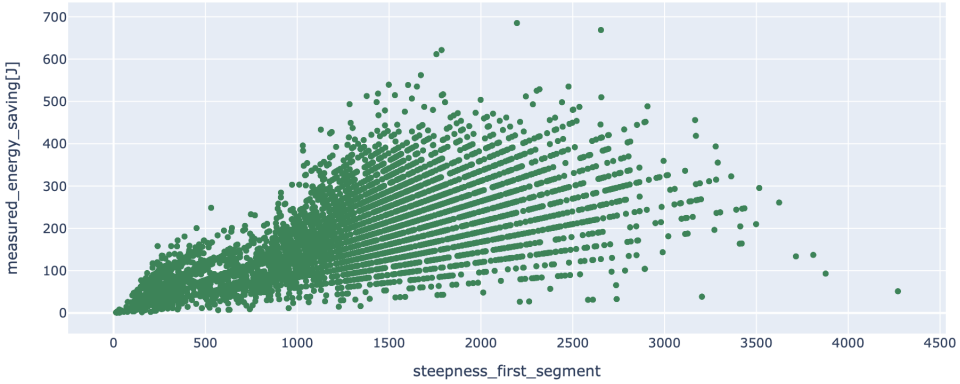


Figure 4.3: Relationship between steepness and measured energy saving

While the absolute reduction in EC is straightforward and generally highly correlated with suitability, a more accurate description of suitability is based on the 'steepness' of the Decreasing Consumption segment. In other words, we consider how rapidly EC decreases with increasing duration, effectively normalizing energy saving with respect to time cost. Although the actual profile generally does not decrease linearly, this approach is sufficiently accurate for assessing the suitability of an optimization candidate. In Figure 4.3, where the x-axis represents the steepness and the y-axis denotes the measured energy saving, it becomes apparent that different points with the same steepness exhibit a wide range of measured energy savings. This variability illustrates that the same suitability can be achieved in multiple ways: we may achieve moderate savings rapidly or substantial savings over a slightly longer duration

or anything in between. One major limitation of the steepness, as proposed, is that it only compares the profiles based on the Decreasing Consumption segment, this is however acceptable as we generally only/more care about this part of the EPs.

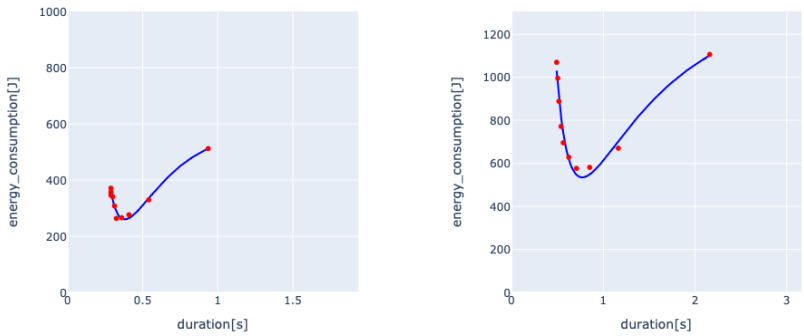


Figure 4.4: Two EPs with same steepness

Figure 4.4, clearly shows that both profiles are different, yet they are equally suitable (with the same steepness) for optimization. To calculate the steepness, we employ the following equation:

$$\text{steepness [J/s]} = \frac{\text{energy saving [J]}}{\text{duration penalty [s]}} \tag{4.1}$$

The energy saving and duration penalty are obtained by calculating the difference between the x and y coordinates of the measurement point with optimal EC and the measurement point at 100 % speed limit. We opt for using the measured optimum instead of an estimated one (from EP) since the granularity of measurements sufficiently accurately captures the optimum.

While the steepness is very useful for comparing EPs against each other, it is not very useful for analysing what features affect the consumption (as seen in Figure 4.5). For that reason we will use the measured energy saving and measured duration penalty for selecting important features. In the Figures 4.5-4.7 we can see the correlation matrices for several features.

First, we start by analysing the static features. While features such as Euclidean, horizontal, and vertical distances all show correlation with measured energy saving, the most significant correlation with energy saving is observed for various joint-related features. The high correlation for feature theta sum (which represents the overall complexity of the movement), suggests that more complex movements generally offer greater potential for energy saving. However, it is noteworthy that this relationship holds true only for the first three joints, namely j_{1-3}, where we observe a positive correlation, especially true for j_2. Surprisingly, for the joints j_{3-4}, there is a negative correlation. To refine the earlier statement, it appears that operations involving substantial movement in the first three joints are the most suitable candidates for optimization. Conversely, complex movements in the last three joints

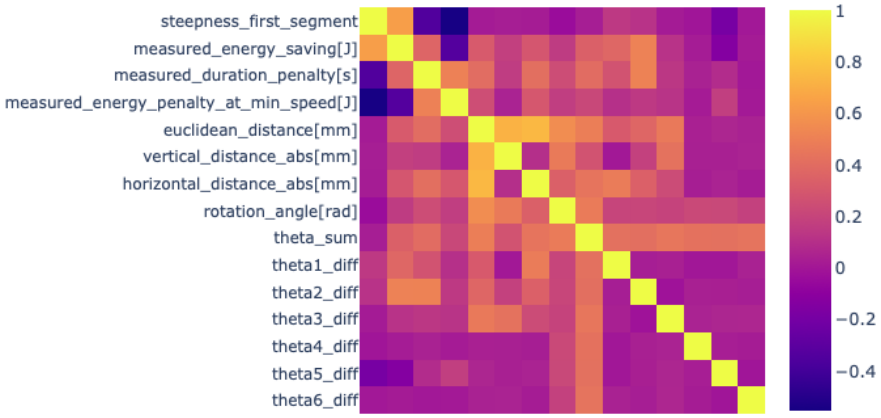


Figure 4.5: Correlation matrix for static features

tend to be less suitable for energy optimization. We believe this disparity is influenced by two main factors. Firstly, the robot's physical structure, where the first three joints typically employ larger motors compared to the last three joints (they carry the robot's weight as well as the manipulated object). Secondly, the movement types associated with the first three joints are generally more energy-demanding compared to those of the last three joints (e.g., long vertical movements, rotational movements in the base, etc.).

Further, when analyzing dynamic attributes like acceleration, deceleration, or speed, comparable patterns emerge. The dynamic attributes of the initial three joints exhibit the most substantial correlation with measured energy saving, with the strongest correlation observed in the second joint (see Figure 4.6). Conversely, the last three joints display small to moderate negative correlation with measured energy saving (see Figure 4.7).

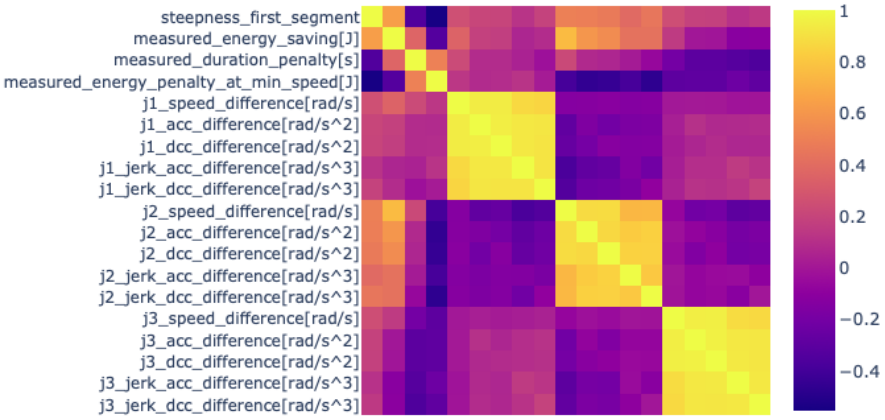


Figure 4.6: Correlation matrix for dynamic features of $j_1 - j_3$

In addition, it's vital to explore the measured duration penalty. When considering static features, the correlation matrix for duration penalty somewhat matches that of energy saving, with distinct distances and configuration differences in the first three joints being the most significant features. However,

when it comes to dynamic features, little to no correlation appears to exist between them and the duration penalty.

Finally, the measured energy penalty at min speed, which represents the disparity in energy consumption at the optimal speed limit and the minimum speed limit (10 %), also demands attention. This value is of interest as it helps characterize the Increasing Consumption segment, which will later be used for profile prediction. The correlation of distance features appears similar but slightly weaker compared to measured energy saving. Intriguingly, the joint features display somewhat contrasting behavior, particularly in the case of dynamic features where the correlation tends to be negative where there's positive correlation for measured energy saving and vice versa.

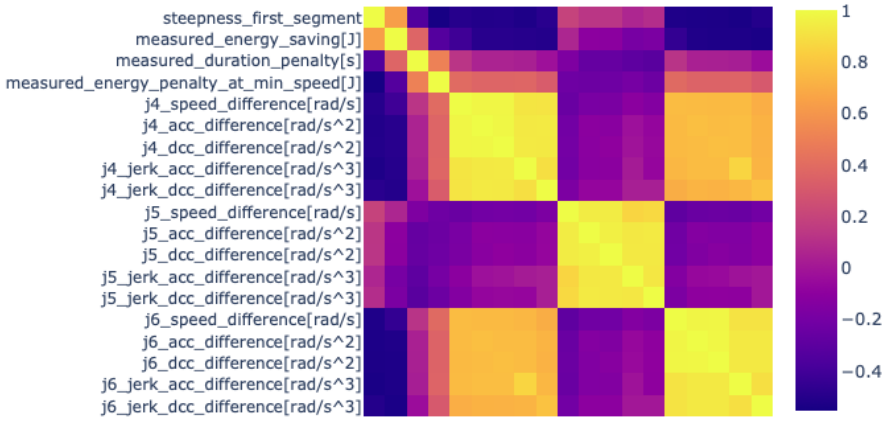


Figure 4.7: Correlation matrix for dynamic features of $j_4 - j_6$

Chapter 5

Energy Profile and Idle Consumption Approximation

In this chapter, we are going to illustrate how we applied the findings from the previous chapter to modify the existing method of obtaining EPs (Section 3.2) and the idle power consumption coefficients (Section 3.3). We will start by revisiting the current process and discussing how we can modify it, we will also share some assumptions about the EC. This is followed by discussing the considered ML models and an evaluation of the accuracy of our selected models. In the end, we will present several predicted EPs to demonstrate the results of our work.

5.1 Estimator Design

As highlighted in Chapter 3, our goal is to modify the method of obtaining the EP within the eRobot plugin. The existing method utilizes the KRC plugin to run a series of simulations at various speed limits to capture consumption data. This data is then interpolated to create a polynomial EP (Figure 4.2). Following this, the polynomial EP is simplified into a set of linear segments (Figure 3.3), which are then incorporated into a MILP model to find the optimal solution.

Let's begin by reconsidering the problem statement. The objective function (Equation 3.2) can be interpreted as maximizing energy saving for a specific CT. Therefore, our main concern is not identifying or approximating the actual EP. We are solely interested in its shape; the baseline or intercept does not matter since it does not affect the optimal solution, but rather only the objective value. In essence, we do not even need the exact shape. Our goal could only be to rank the profiles from most to least suitable and greedily reduce the speed limits until we achieve the desired CT. Therefore, the precision of our prediction in comparison to the actual profile is not our primary concern. What matters more is ensuring that when comparing two actual EPs with their predicted counterparts, the order of suitability remains consistent. Furthermore, it is important to realise that we maintain the ability to accurately determine the duration of any movement at any speed limit, as well as access to data on joint speeds, accelerations, and so on. The only

simulation output we aim to replace is the EC estimate.

Now that we have identified the objective, the next challenge is identifying the most appropriate point to intervene in the current process. One approach could be to supply the eRobot plugin with the polynomial EP, from which the plugin could then generate a linear approximation. On the other hand, we could directly deliver the linear approximation, eliminating the need to create a polynomial EP in the first place. Furthermore, we also need to estimate the idle power consumption (Section 3.3), which is currently derived by averaging the final few consumption measurements before the PtP destination is reached.

From our initial experiments, we quickly realized that attempting to directly predict the polynomial EP (i.e., predicting the coefficients) is impractical. This is due to two key reasons. Firstly, the polynomial is already an approximated representation of the actual relationship, which inherently introduces discrepancies into the training data, resulting in two potential sources of error, making it harder to fine-tune. Secondly, the nature of the chosen polynomial is a concern. The polynomial typically exhibits convex behavior within the 'interval of interest' (spanning from the minimum duration at 100 % speed to the maximum duration at 10 % speed). However, outside of this range, the function often behaves unpredictably, displaying concave segments. Misestimating certain coefficients could shift these concave regions within the interval of interest, leading to an unexpected profile behavior. Moreover, two profiles that appear very similar (overlapping EPs) within the interval of interest often display significant differences outside of this interval, thereby complicating the task of generalization.

Instead, we chose to supply the plugin with already linearly approximated profile with a fixed shape. As shared in previous chapter (Section 4.3), we can distinguish three different regions of the profile, namely the Decreasing, Near-Constant and the Increasing Consumption segments. We could model the profile so that each of these segments is approximated by single line. The profile would then be defined by a function (similar to approximation in [3]):

$$E(d) = \begin{cases} -a_1 \cdot d + b_1, & \text{if } d \in (d_{min}, d_1) \quad a_1, b_1 \geq 0 \\ const., & \text{if } d \in (d_1, d_2) \\ a_2 \cdot d - b_2, & \text{if } d \in (d_2, d_{max}) \quad a_2, b_2 \geq 0 \end{cases} \tag{5.1}$$

To put it simply, the energy profile is approximated by a decreasing linear function over the range between the fastest duration and the duration $d_1 = d_{opt} - \delta_1(d_{min}, d_{opt})$. Here, d_1 is the left neighbourhood of the duration that results in optimal EC – d_{opt}. Then, there is a constant function (which could be zero) on the interval between d_1 and $d_2 = d_{opt} + \delta_2(d_{opt}, d_{max})$, d_2 being the right neighbourhood of d_{opt}. This is followed by an increasing linear function between d_2 and the maximum duration (10 % speed limit). Importantly, this function is fully described with only three pieces of information: the optimal duration d_{opt}, the measured energy saving and the measured energy penalty at min speed, therefore we need to construct three predictors, one for each of

these values, using the findings from previous chapters. The values δ_1 and δ_2 are both function of respective duration values.

Lastly, we need to consider the idle EC. Initially, we considered predicting the value in the same manner as other variables. However, after reviewing data from KRC, we started questioning its precision in estimating static consumption. This is because we consistently observed similar EC values (around 430-450W), irrespective of the TCPF's position. Our approach then shifted, proposing that the idle EC might actually be equivalent to the EC of a movement at an incredibly slow speed limit, which can be inferred from the profile, essentially corresponding to the slope of the profile at infinity (although this is not entirely accurate as the profile represents the entire movement, hence the tangent would be some kind of average of all idle ECs along the trajectory). However, as mentioned in Section 4.3, the EC in the Increasing Consumption segment can be precisely approximated by a straight line throughout its entire length. Consequently, we assigned the static consumption to be the slope of this segment.

5.2 Estimator Implementation

As concluded in the previous section, we need to predict free values, one each for the optimal duration – $\hat{d}_{opt}$, the measured energy saving – $\hat{EC}_{d_{min}}$, and the measured energy penalty at the minimum speed – $\hat{EC}_{d_{max}}$. All three of these variables are continuous and bounded. We experimented with two different models, a neural network and a random forest regressor. We eventually chose the random forest regressor as it demonstrated similar accuracy to the neural network but was simpler to implement and quicker to train. For all our predictors, we used the same input features, specifically, the vertical and horizontal distance (we did not use Euclidean distance as the information it carries is already included in the two chosen distances), the rotation angle, the differences in the θ_{1-6} angles, and all the j_{1-6} dynamic features. Prior to feeding the features to models, we used a standard scaler - removing the mean and dividing the values by standard deviation. This process ensures that all features have a mean of 0 and a standard deviation of 1.

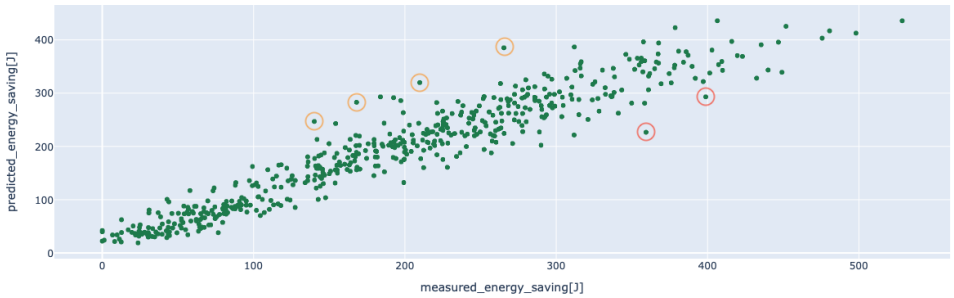


Figure 5.1: Predicted energy saving compared to measured energy saving

Now, let's take a look at the results we have obtained. The accuracy of predicted values is listed in Table 5.1[1] and visualised in Figures 5.1 – 5.3.

	e_{abs}				e_{rel}			
	25 %	50 %	75 %	max	25 %	50 %	75 %	max
$\hat{EC}_{d_{min}}$	7.99	17.38	33.64	132.72	4.5	10.4	20.7	246.6
$\hat{d}_{opt}$	0.014	0.029	0.047	0.280	2.3	4.5	8.4	24.7
$\hat{EC}_{d_{max}}$	11.48	23.18	40.41	220.27	2.3	4.8	8.4	67.8

Table 5.1: Relative and Absolute errors of predictors

The first value we predicted is the energy saving. Even though the average relative error $e_{rel} = \frac{|predicted - measured|}{measured}$ is somewhat high, around 18.5 % (largely impacted by few high values), the main function of the predicted energy saving is, in a sense, to rank the EPs from the best to the worst. Therefore, our main concern revolves around the outliers (which are shown in red - underpredicted and orange - overpredicted in Figure 5.1). Essentially, it's crucial to avoid labeling a good (large steepness) profile as a bad one, and even more so, misidentifying a bad profile as a good one. This is because it changes the order in which the profiles are chosen during optimization. Mistakenly ordering two profiles with a similar real steepness then results only in a slightly sub-optimal solution.

The second value we predicted – the optimal duration, also impacts the steepness (Equation 4.1). However, mispredicting this value can actually lead not only to sub-optimal energy saving, but to an increase in EC. Consider a scenario where the predicted duration would lay in the Increasing Consumption segment of the actual EP. In this case, we could potentially end up with a higher EC than without optimization. Moreover, we would also ineffectively use up the available time that could have been utilized to save energy on a different movement. The average relative error for this value is slightly less than 6 %, with the vast majority of values falling within ±10 % of the true value. The test results are plotted in Figure 5.2.

Figure 5.2: Predicted optimal duration compared to measured optimal duration

[1] large e_{rel} typically occur for small values - e.g. $\hat{EC}_{d_{min}} = 60J$ vs. $EC_{d_{min}} = 20J$, large e_{abs} are common for large values - e.g. $\hat{EC}_{d_{min}} = 420J$ vs. $EC_{d_{min}} = 550J$

Finally, the mean relative error for predicting the energy penalty at minimum speed is even lower, slightly above 5 % (see Figure 5.3 for test results). While this part of the profile is not as important in the context of movement optimization (we only really use this segment when there are significant differences between the unoptimized CT and the target CT), we still need this value to calculate the idle consumption coefficients (Section 3.3).

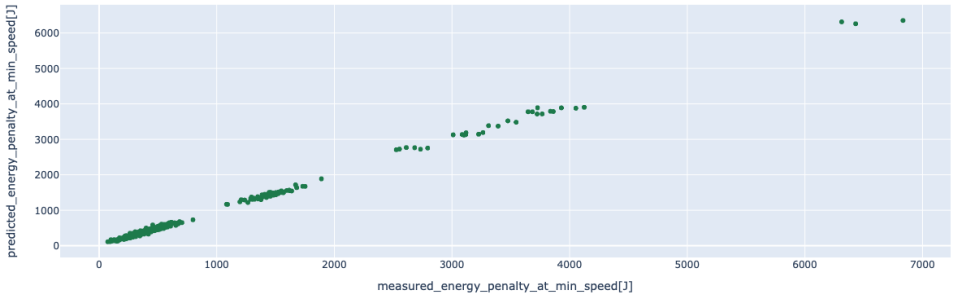


Figure 5.3: Predicted energy penalty at min speed compared to measured energy penalty at min speed

To sum it all up, we can now substitute these predicted values into the proposed approximation formula (Equation 5.11). The coefficients in $\delta_{1,2}$ were set empirically, precisely to 40 % and 4 %, the y-offset is arbitrarily set at 50J (this value does not influence the optimization result, only the objective value). The EP is then obtain using the following equations:

$$c = 50 \tag{5.2}$$

$$\delta_1(d_{\min}, \hat{d}_{opt}) = 0.4 \cdot (\hat{d}_{opt} - d_{\min}) \tag{5.3}$$

$$\delta_2(\hat{d}_{opt}, d_{\max}) = 0.04 \cdot (d_{\max} - \hat{d}_{opt}) \tag{5.4}$$

$$d_1 = \hat{d}_{opt} - \delta_1 \tag{5.5}$$

$$d_2 = \hat{d}_{opt} + \delta_2 \tag{5.6}$$

$$a_1 = \frac{-\hat{EC}_{d_{min}}}{\hat{d}_{opt} - d_1 - d_{min}} \tag{5.7}$$

$$a_2 = \frac{\hat{EC}_{d_{max}}}{d_{max} + d_2 - \hat{d}_{opt}} \tag{5.8}$$

$$b_1 = -a_1 \cdot (\hat{d}_{opt} - d_1) + c \tag{5.9}$$

$$b_2 = -a_2 \cdot (\hat{d}_{opt} + d_2) + c \tag{5.10}$$

$$E(d) = \begin{cases} a_1 \cdot d + b_1, & \text{if } d \in (d_{\min}, d_1) \\ c, & \text{if } d \in (d_1, d_2) \\ a_2 \cdot d + b_2, & \text{if } d \in (d_2, d_{\max}) \end{cases} \tag{5.11}$$

In Figure 5.4, we provide several examples of such predicted EPs. The original measurements were adjusted with an offset to align at 50J.

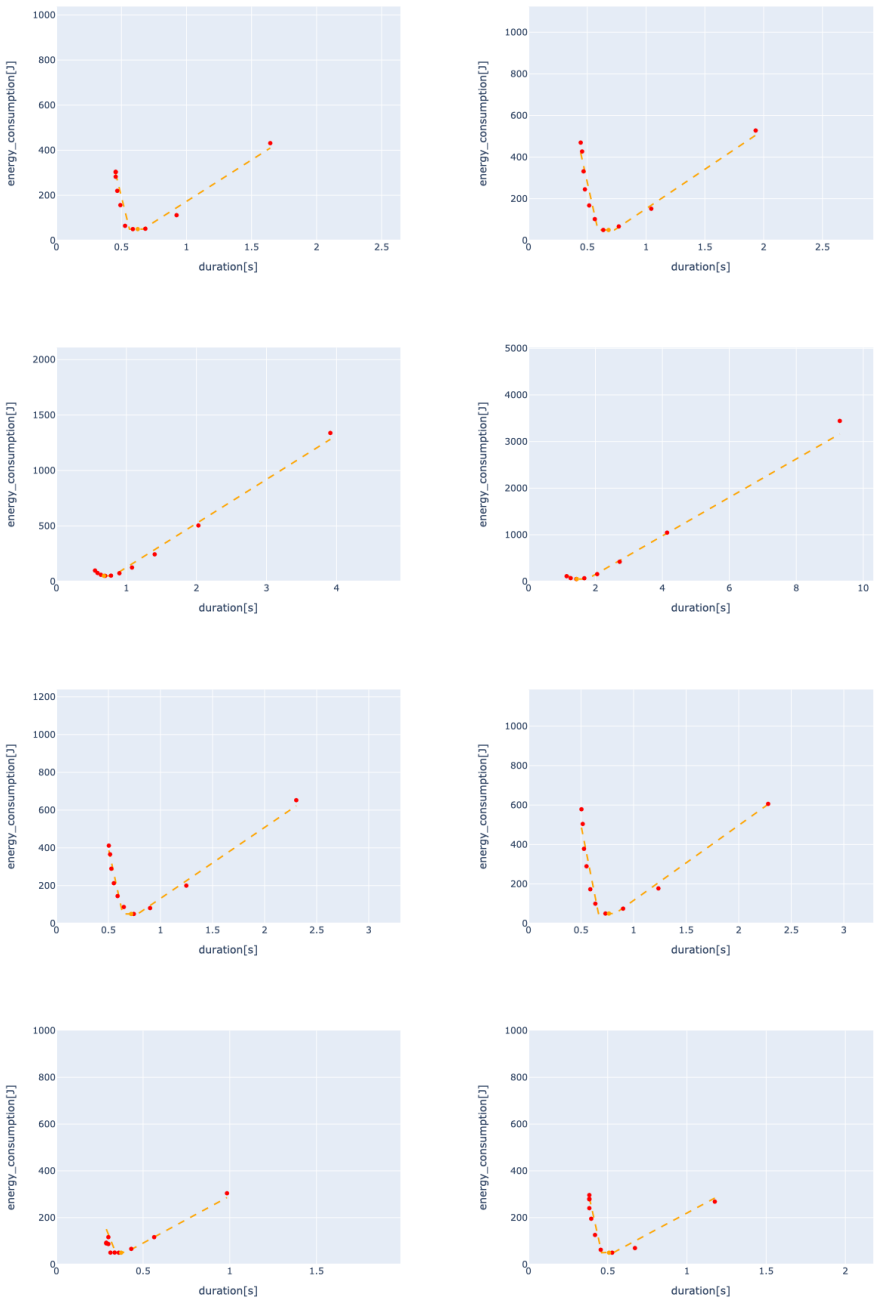


Figure 5.4: Predicted profiles and original measurements offset to overlap

Chapter 6

Experimental Results

This chapter presents and interprets the outcomes obtained from using the optimization algorithm on a generated dataset with our predicted EPs. We then compare these findings with results from the non-optimized OT and results of optimization using the original EPs sourced from KRC.

6.1 Experimental Setup

During the testing phase, we encountered same challenges as discussed in Section 4.1, namely, the shortage of real-life simulations to conduct extensive tests. Thus, we adopted identical method as described in Algorithm 1 to generate approximately 100 random OTs. We adjusted the budget parameter B to generate a wide array of movements, ranging from simple to very complex ones. From this generated dataset, we handpicked 30 OTs – 10 each from simple, medium, and complex movements – for test runs. The categorization into simple, medium, and complex is not strictly quantifiable, but based on a visual classification of the movements. The purpose of this selection was to ensure a comprehensive evaluation of our approach across diverse scenarios. The selected 30 OTs were further divided into three subsets, each roughly balanced in terms of movement complexity. Each subset was then subjected to a slightly different simulation setup.

For the first two setups, we began by running the simulation at 100 % speed, noting down the $EC_{d_{min}}$ and the duration d_{min}. Subsequently, two optimization runs were carried out for each target CT set at 110 %, 121 %, and 133 % of d_{min} (these CTs were chosen arbitrarily). One run utilized profiles obtained using KRC, while the other one employed the predicted profiles. The only difference between these two setups is the number of linear segments used to approximate the polynomial profile in the optimization with KRC. This is an input parameter in the KRC plugin (Subsection 3.2.1). For the first subset, we used four segments, and for the second subset, we increased the number of segments to six.

For the final setup, we adjusted the target CTs to be 150 %, 175 %, and 200 % of the minimum duration d_{min}. Similar to the first setup, we used four linear segments for the approximation of the polynomial profile in the optimization process with KRC.

In all three setups, the unoptimized EC for a given CT was calculated using the following formula:

$$EC_{d_{min}+wait} = EC_{d_{min}} + (CT - d_{min}) \cdot 430 \text{ W} \tag{6.1}$$

This assumes that the robot's EC during its idle state is a constant value of 430 W. We can apply the same idle consumption to all OTs because the robot always returns to the same location at the end – $HOME$ (recall Section 4.1). We then ran both optimized OTs and used KRC to obtain the final EC.

6.2 Results

As stated in the preceding section, a total of 30 distinct simulations were executed. We then ran two simulations for each CT – one with KRC profiles, one with predicted profiles. To compare these two methods against each other we define the relative energy saving achieved compared to unoptimized EC:

$$relative\ energy\ saving = \frac{EC_{d_{min}+wait} - EC_{opt}}{EC_{d_{min}+wait}}$$

The average relative savings for each method are shown in Table 6.1. From the data presented in the table, we can deduce two key insights. First, irrespective of the EP utilized in the optimization – whether it's the original one procured from KRC or the new, predicted one – significant energy savings are realized in comparison to the unoptimized OTs. Second, when considering shorter CTs relaxations (up to 133 %), the optimization leveraging the newly predicted profile surpasses the performance of the original methodology. This holds true for both the 4-segment linear approximation as well as the 6-segment approximation of the KRC profile. However, this edge diminishes for larger CTs, where the traditional optimization proves more effective than our novel approach.

	4 segments		6 segments	
d_{min} [%]	KRC [%]	pred. [%]	KRC [%]	pred. [%]
110	13.37	17.42	12.75	17.62
121	24.14	28.19	25.93	32.36
133	32.76	34.10	37.26	38.07
150	35.89	35.81	–	–
175	43.06	41.04	–	–
200	49.64	47.17	–	–

Table 6.1: Average relative energy saving

Upon a more detailed examination of the results, the superior performance of our predicted profiles compared to the original method initially appears unexpected. One might assume that the KRC profile, having a presumably more direct source of information, would inherently be more accurate than our predictions. Generally speaking, while the polynomial EPs indeed tend to be more accurate (yet still approximation) than our predicted ones, the real issue emerges when we consider their linear approximation, as illustrated in Figure 6.1. Green line is the original polynomial profile and red lines represent the linear approximation used in MILP.

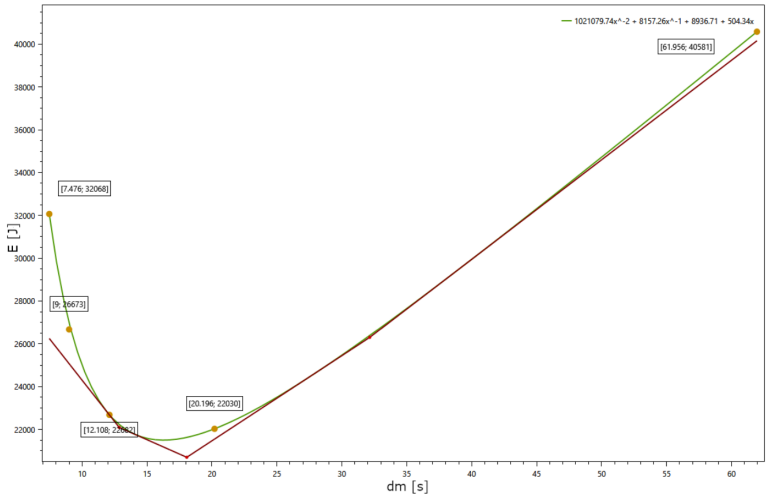


Figure 6.1: Linearly approximated EP with 4 segments used in KRC optimization [3]

Most notably, the Decreasing Consumption segment is often inadequately represented by the linear approximation, which affects the order in which operations are chosen for the optimization. Given that in the context of a single IR, the MILP essentially prioritizes profiles in a sort of 'greedy' slowdown mechanism, ordered by their steepness. This linear approximation imprecision can therefore lead to incorrect order and less-than-optimal energy conservation outcomes. Additionally, upon closer inspection, we observe that the second linear segment can also exhibit a declining trend with a relatively steep gradient. This gradient, unfortunately, does not align closely with the true nature of the actual profile. As seen in the figure, the steepness of the second segment is almost comparable to that of the first segment. This in turn may prompt the optimization to favor further modifying operations within the Near-Constant Consumption segment rather than advancing to a subsequent operation with less steep Decreasing Consumption segment. This pattern is evident in Figure 6.2 – the OT optimized using the KRC profile (Figure 6.2a) leans towards decelerating a limited number of profiles more significantly. In contrast, the optimization that employs the predicted profiles (Figure 6.2b) tends to moderate more movements to a lesser extent.

For extended CTs, this imperfection does not present a significant problem because we generally reach the optimal speed limit for all movements –

Paths & Locations	Speed	Acc	Motion	Zone	Duration
Test_optimized_random_0_optim...					
Dummy					5.45
HOME	100 %	100 % (def)	PTP	FINE	0.16
via0	52 %	100 % (def)	PTP	FINE	0.43
via1	100 %	100 % (def)	PTP	FINE	0.6
via2	35 %	100 % (def)	PTP	FINE	0.67
via3	100 %	100 % (def)	PTP	FINE	0.4
via4	33 %	100 % (def)	PTP	FINE	0.58
via3	31 %	100 % (def)	PTP	FINE	0.61
via2	100 %	100 % (def)	PTP	FINE	0.41
via1	36 %	100 % (def)	PTP	FINE	0.66
via0	100 %	100 % (def)	PTP	FINE	0.6
HOME	100 %	100 % (def)	PTP	FINE	0.34

(a) : Using KRC profile

Paths & Locations	Speed	Acc	Motion	Zone	Duration
Test_optimized_random_0_optim...					
Dummy					5.46
HOME	100 %	100 % (def)	PTP	FINE	0.16
via0	60 %	100 % (def)	PTP	FINE	0.4
via1	70 %	100 % (def)	PTP	FINE	0.73
via2	46 %	100 % (def)	PTP	FINE	0.58
via3	63 %	100 % (def)	PTP	FINE	0.48
via4	49 %	100 % (def)	PTP	FINE	0.47
via3	49 %	100 % (def)	PTP	FINE	0.47
via2	60 %	100 % (def)	PTP	FINE	0.49
via1	46 %	100 % (def)	PTP	FINE	0.59
via0	74 %	100 % (def)	PTP	FINE	0.71
HOME	60 %	100 % (def)	PTP	FINE	0.4

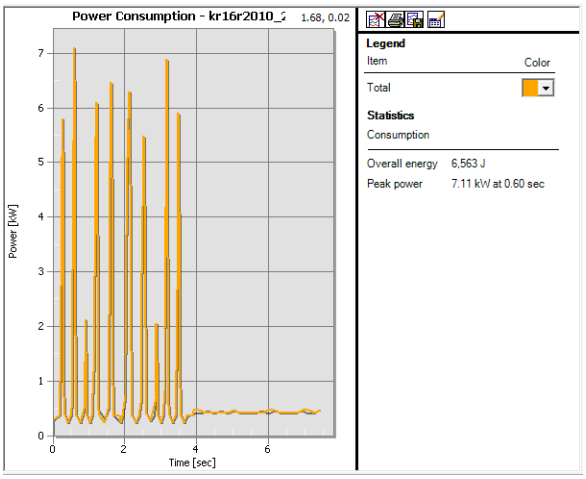
(b) : Using predicted profile

Figure 6.2: Optimized OTs

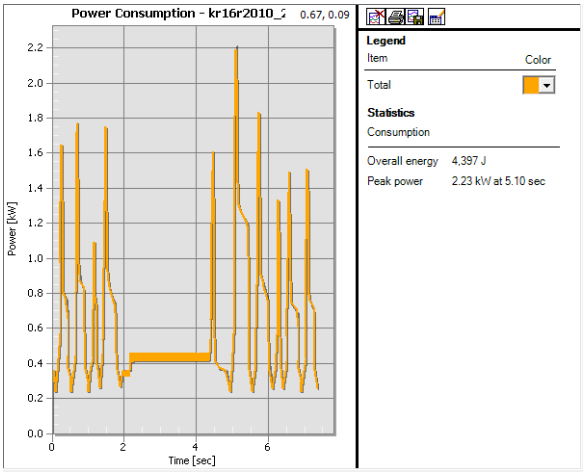
optimally reducing all speed limits in 'incorrect' order, still converges to the optimal solution. Furthermore, for very large CTs, we often traverse the entire Near-Constant segment for all the movements and then the consumption starts rising again in the Increasing Consumption segment. This is where we start utilizing the idle consumption, and the predicted profile starts lagging behind the KRC profiles. Relying on a tangent at infinity as a proxy for idle consumption seems to introduce certain degree of inaccuracy, resulting in seemingly lower EC compared to the actual estimate[1] from KRC, in turn leading to sub-optimal insertion of wait times, as seen in Figure 6.3 – an example of optimized OT where target CT is roughly twice the original one. In the figures, we can see the consumption graphs exported from KRC, first for unoptimized OT (we added the wait time at the end to meet the CT) and then the two optimized ones. Notice the pronounced wait time (around 2 seconds) in the middle of the execution for predicted profile OT compared to much shorter (less than a second) at the beginning of the KRC profile OT.

Notably, the time required to obtain the predicted profiles for tests in in Figure 6.3 was nearly halved compared to obtaining KRC profiles, as we now

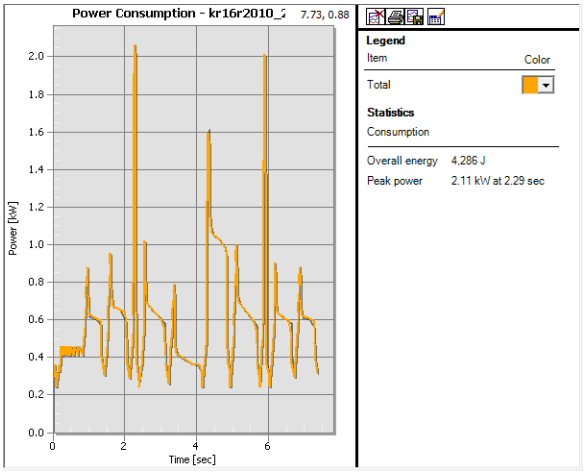
[1] we harbor some reservations regarding the precision of the idle consumption estimation from KRC (see Section 3.3)



(a) : Unoptimized OT



(b) : Optimized OT – predicted EP



(c) : Optimized OT – KRC EP

Figure 6.3: Random experiment – CT $\approx 2 \cdot d_{min}$

execute only 2 initial simulations compared to the original 5. This reduction is related solely to the creation of the EPs and not the entire optimization process. Nevertheless, the optimization itself typically runs more swiftly, which can be attributed to the fewer constraints in the MILP formulation, determined by the number of linear segments.

Delving deeper, we can see the detailed results from each simulation, presented in Table 6.2 for the initial two sets of experiments, and in Table 6.3 for the concluding set. In these tables, each row displays the unoptimized CT, labeled as d_{min}, alongside the EC for the unoptimized OT. Additionally, the tables provide the total consumption, including idle times, for every specified target CT. Each CT is accompanied by two optimized consumption values: the first derived from the KRC-sourced profiles, and the second based on our predicted profiles.

The initial table reveals that enhancing the segment count positively influences optimization using the KRC profiles. This is evident from the higher number of solutions that surpass the results from predicted profiles with the 6-segment approximation compared to the 4-segment approximation. However, this improvement is offset by longer optimization durations. Nevertheless, regardless of the method, there is a notable improvement in EC compared to unoptimized version.

The subsequent table proves that when the optimization does not heavily rely on wait times (minor relaxation on CT), the predicted profiles either surpass or match the performance of the KRC profiles. However, this advantage diminishes as the optimization begins to incorporate extended wait durations, as illustrated in Figure 6.3b. In these cases, the KRC consistently exceeds the performance of the predicted profiles, owing to its more precise wait consumption estimation and the accurate approximation of Increasing Consumption segment in both methods.

6.3 Summary

We examined the efficiency of the optimization algorithm on a generated dataset using both the predicted and the original EPs sourced from KRC. The experiments consisted of 30 tests (OTs) with varying movement complexity. Simulations showed significant energy savings for both EPs compared to unoptimized OTs, ranging from 13 % to 50 %. Intriguingly, optimization with the newly predicted profiles outperformed the original method for shorter cycle times (up to 133 %). However, for more extended CTs, traditional optimization using the KRC profiles was more effective. This performance difference is attributed to the linear approximation of the KRC profiles, which may lead to less-than-optimal energy conservation for smaller CTs relaxations, and the inefficiencies introduced by the idle consumption prediction when extended wait durations are incorporated in larger CTs relaxations. The use of predicted profiles, however, had the advantage of nearly halving the time required for obtaining profiles compared to using KRC.

#	d_{min}[s]	$EC_{d_{min}}$[J]	$EC_{d_{min}+wait}$[J] +10%	+21%	+33%	EC_{opt}[J] +10% KRC	+10% pred.	+21% KRC	+21% pred.	+33% KRC	+33% pred.
						4 segments					
1	5,8	8689	8938	9462	10288	7587	**7244**	6890	**6688**	6937	**6718**
2	5,12	8652	8872	9334	10063	7755	**7298**	7044	**6603**	6806	**6519**
3	7,76	10464	10798	11498	12603	9453	**9352**	9422	**9235**	9364	**9307**
4	7,61	9396	9723	10410	11494	8408	**8215**	8287	**8246**	8406	**8300**
5	4,51	7093	7287	7694	8336	6223	**5964**	5778	**5182**	5325	**5142**
6	5,3	6784	7012	7490	8245	6092	**5988**	**5706**	5738	**5785**	5895
7	4,52	5948	6142	6551	7194	5246	**5058**	5053	**4864**	4956	**4900**
8	3,64	5978	6135	6463	6981	5376	**5000**	4734	**3956**	4184	**3871**
9	4	5000	5172	5533	6103	4482	**4258**	4059	**3974**	4029	**3996**
10	4,36	6698	6885	7279	7900	6062	**5368**	5370	**4800**	4820	**4791**
						6 segments					
11	4,93	7453	7665	8110	8812	6917	**6205**	6047	**5478**	5562	**5510**
12	5,27	8870	9097	9572	10323	8015	**7408**	7204	**6362**	**6209**	6242
13	4,86	8077	8286	8725	9417	7405	**6797**	6458	**5772**	5975	**5754**
14	4,42	6910	7100	7499	8128	**6012**	6032	5287	**5188**	**5120**	5219
15	4,68	7777	7978	8401	9067	6971	**6465**	6345	**5547**	5638	**5547**
16	4,79	7486	7692	8125	8806	6801	**6523**	6121	**5509**	5522	**5392**
17	5,16	7248	7470	7936	8670	**6580**	6679	**6173**	6223	6189	**6175**
18	3,37	4425	4570	4874	5354	3935	**3707**	3465	**3169**	**3068**	3161
19	3,68	4591	4749	5082	5605	4073	**3810**	3819	**3198**	3348	**3241**
20	3,49	3968	4118	4433	4930	3476	**3204**	3166	**2963**	3176	**2981**

Table 6.2: Simulation results – 110-133 % d_{min}

			$EC_{d_{min}+wait}$[J]			EC_{opt}[J]					
			+50%	+75%	+100%	+50%		+75%		+100%	
#	d_{min}[s]	$EC_{d_{min}}$[J]				KRC	pred.	KRC	pred.	KRC	pred.
4 segments											
21	4,08	6388	7265	8581	10335	4543	**4413**	**4732**	4851	**5126**	5289
22	5,09	9238	10332	11974	14163	**5996**	6068	**6286**	6548	**6751**	7109
23	5,27	8323	9456	11156	13422	6056	**5933**	**6256**	6493	**6545**	7043
24	4,26	7763	8679	10053	11885	**4599**	4626	**4909**	5107	**5312**	5565
25	4,21	6269	7174	8532	10342	4530	**4393**	**4663**	4840	**5090**	5276
26	3,71	5011	5809	7005	8600	**3510**	3622	**3877**	4009	**4300**	4410
27	5,71	6859	8087	9928	12383	6615	**6535**	**7026**	7137	**7615**	7736
28	3,95	5926	6775	8049	9748	**3925**	3988	**4265**	4414	**4691**	4832
29	4,09	4986	5865	7184	8943	**4576**	4642	**4775**	5064	**4775**	5496
30	3,54	4630	5391	6533	8055	**3351**	3448	**3697**	3804	**4078**	4179

Table 6.3: Simulation results – 150-200 % d_{min}

Chapter 7

Conclusion

The thesis aimed to change the way we obtain energy profiles in the eRobot plugin. Instead of using KRC estimates, we explored machine learning methods to predict these profiles. By doing this, we hope to broaden the compatibility of the plugin, making it applicable not only to KUKA systems but to a diverse range of vendors in the industry.

To start, we familiarized ourselves with the energy profile concept presented by Gadaleta et al. [12]. We also explored Process Simulate and the eRobot plugin [3], [17].

To gain a deeper understanding of the profiles, we utilized a modified version of the plugin to produce thousands of random energy profiles. Upon analyzing these, we discerned that the linear approximation of the profiles typically comprises three distinct segments: a Decreasing Consumption segment, a Near-Constant segment, and an Increasing Consumption segment. Notably, these segments can be characterized by just three parameters: energy saving, optimal duration, and the energy penalty at minimum speed. Based on our findings, we developed three random forest predictors for these parameters, using the most influential features identified during analysis. We subsequently employed these predictors to obtain the linear approximation of the profiles, and integrated these approximations back into the plugin.

We conducted 30 random simulations of varying movement complexities to compare the optimized consumption using the new method against both the unoptimized consumption and the optimized consumption derived from the existing KRC profiles. We ran the optimizations for a range of relaxed target cycle times, from 110 % to 200 % of the initial cycle time. The findings revealed that our new approach surpassed the original method for modest relaxations. This superiority was linked to the manner in which the original method formed its linear approximation. However, for extended cycle times, the new method was slightly less effective, a shortfall attributed to imprecise predictions of idle consumption. Regardless, both techniques yielded energy savings ranging roughly between 13 % and 50 %.

7.1 Future Work

While our study has achieved promising results, there are two areas we've identified for further refinement to enhance its broader application.

The primary constraint is our reliance on a single robot, the KUKA KR16R2010, throughout our research. This was due to challenges in obtaining and setting up the digital twins of industrial robots. While this doesn't entirely negate the potential applicability to other robots, it significantly impacts the solution's accuracy. To enhance accuracy, access to a broader range of robots, either virtual or physical, is essential. Preferably, these robots should come from various vendors and have different sizes. Notably, there's evidence suggesting that robots of similar sizes from different manufacturers might exhibit comparable energy consumption patterns. To optimize the plugin's efficacy, we could either train individual predictors for each robot type or develop a more universal model to predict energy usage across different robot sizes.

Second limitation arises from our exclusive use of the *fine* zone setting for point-to-point movements. This setting dictates how close the robot must get to its destination for the movement to be deemed complete. Crucially, the plugin employs this setting internally to cluster several operations that would then be described by a singular energy profile. By limiting the setting to *fine*, we ensured that the profile is always linked to a single operation. Incorporating other zone settings would necessitate adjustments to either the plugin or the prediction.

Bibliography

[1] IEA, "Electricity Information: Overview – Analysis," Tech. Rep. [Online]. Available: https://www.iea.org/reports/electricity-information-overview/electricity-consumption

[2] P. Šůcha and A. Pochylý, "Optimalizace výrobního taktu a spotřeby robotických buněk." [Online]. Available: https://www.konference-roboty.cz/

[3] M. Petr, "Modely pro minimalizaci spotřeby robotických buněk," Jun. 2019. [Online]. Available: https://dspace.cvut.cz/handle/10467/83340

[4] bp Statistical Review of World Energy, "The 2022 bp Statistical Review of World Energy," Tech. Rep. [Online]. Available: https://www.bp.com/content/dam/bp/business-sites/en/global/corporate/pdfs/energy-economics/statistical-review/bp-stats-review-2022-full-report.pdf

[5] F. Kuik, J. F. Adolfsen, A. Meyler, and E. Lis, "Energy price developments in and out of the COVID-19 pandemic – from commodity prices to consumer prices," Jun. 2022. [Online]. Available: https://www.ecb.europa.eu/pub/economic-bulletin/articles/2022/html/ecb.ebart202204_01~7b32d31b29.en.html

[6] D. Meike and L. Ribickis, "Energy efficient use of robotics in the automobile industry," in *2011 15th International Conference on Advanced Robotics (ICAR)*, Jun. 2011, pp. 507–511.

[7] M. Bottin, G. Rosati, and G. Boschetti, "Working Cycle Sequence Optimization for Industrial Robots," in *Advances in Italian Mechanism Science*, V. Niola and A. Gasparetto, Eds. Cham: Springer International Publishing, 2021, vol. 91, pp. 228–236. [Online]. Available: http://link.springer.com/10.1007/978-3-030-55807-9_26

[8] F. Wang, Z. Wu, and T. Bao, "Time-Jerk optimal Trajectory Planning of Industrial Robots Based on a Hybrid WOA-GA Algorithm," *Processes*, vol. 10, no. 5, p. 1014, May 2022. [Online]. Available: https://www.mdpi.com/2227-9717/10/5/1014

[9] F. Stuhlenmiller, S. Weyand, J. Jungblut, L. Schebek, D. Clever, and S. Rinderknecht, "Impact of Cycle Time and Payload of an Industrial Robot on Resource Efficiency," *Robotics*, vol. 10, no. 1, p. 33, Feb. 2021. [Online]. Available: https://www.mdpi.com/2218-6581/10/1/33

[10] L. Wang, A. Mohammed, X. V. Wang, and B. Schmidt, "Energy-efficient robot applications towards sustainable manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 31, no. 8, pp. 692–700, Aug. 2018. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/0951192X.2017.1379099

[11] M. Chemnitz, G. Schreck, and J. Kruger, "Analyzing energy consumption of industrial robots," in *ETFA2011*. Toulouse, France: IEEE, Sep. 2011, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/6059221/

[12] M. Gadaleta, M. Pellicciari, and G. Berselli, "Optimization of the energy consumption of industrial robots for automatic code generation," *Robotics and Computer-Integrated Manufacturing*, vol. 57, pp. 452–464, Jun. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0736584518301856

[13] M. Gadaleta, G. Berselli, M. Pellicciari, and F. Grassia, "Extensive experimental investigation for the optimization of the energy consumption of a high payload industrial robot with open research dataset," *Robotics and Computer-Integrated Manufacturing*, vol. 68, p. 102046, Apr. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S073658452030257X

[14] Y. Liu, L. Liang, H. Han, and S. Zhang, "A Method of Energy-Optimal Trajectory Planning for Palletizing Robot," *Mathematical Problems in Engineering*, vol. 2017, pp. 1–10, 2017. [Online]. Available: https://www.hindawi.com/journals/mpe/2017/5862457/

[15] A. Mohammed, B. Schmidt, L. Wang, and L. Gao, "Minimizing Energy Consumption for Robot Arm Movement," *Procedia CIRP*, vol. 25, pp. 400–405, 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212827114010865

[16] S. Bjorkenstam, D. Gleeson, R. Bohlin, J. S. Carlson, and B. Lennartson, "Energy efficient and collision free motion of industrial robots using optimal control," in *2013 IEEE International Conference on Automation Science and Engineering (CASE)*. Madison, WI, USA: IEEE, Aug. 2013, pp. 510–515. [Online]. Available: https://ieeexplore.ieee.org/document/6654025/

[17] L. Bukata, P. Šůcha, and Z. Hanzálek, "Optimizing energy consumption of robotic cells by a Branch & Bound algorithm," *Computers & Operations Research*, vol. 102, pp. 52–66, Feb. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0305054818302533

[18] Paryanto, M. Brossog, J. Kohl, J. Merhof, S. Spreng, and J. Franke, "Energy Consumption and Dynamic Behavior Analysis of a Six-axis Industrial Robot in an Assembly System," *Procedia CIRP*, vol. 23, pp. 131–136, 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212827114011482

[19] M. Zhang and J. Yan, "A data-driven method for optimizing the energy consumption of industrial robots," *Journal of Cleaner Production*, vol. 285, p. 124862, Feb. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0959652620349064

[20] M. Grieves, "Origins of the Digital Twin Concept," 2016. [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.26367.61609