



Eötvös Loránd University  
Faculty of Informatics

# Optimizing thread operation on CNC lathe machines

*Author:*

Abdelrahman Moawad  
Computer Science MSc

*Supervisors:*

Gergely Bencsik PhD  
Associate professor

Matyas Ando PhD  
Associate professor

**Szombathely, 2024**

# CONTENTS

|                                                                       |    |
|-----------------------------------------------------------------------|----|
| NOTATIONS.....                                                        | 4  |
| 1. INTRODUCTION .....                                                 | 5  |
| 1.1.1. Problem.....                                                   | 5  |
| 1.1.2. Aims and Overview .....                                        | 6  |
| Mechanical Engineering Perspective.....                               | 6  |
| IT Framework.....                                                     | 7  |
| 2. LITERATURE REVIEW .....                                            | 8  |
| 2.1.1. Industry 4.0 .....                                             | 8  |
| 2.1.2. Different Communication Protocols for CNC .....                | 10 |
| 2.1.3. Threading Operations.....                                      | 12 |
| Existing solutions for thread operations .....                        | 14 |
| 3. Mechanical Engineering Perspective.....                            | 18 |
| 3.1.1. Calculation of the thread area .....                           | 18 |
| 3.1.2. Thread Area of different methods.....                          | 20 |
| 3.1.3. Effect of decreasing number of passes with constant area ..... | 26 |
| 3.1.4. Enhanced function for equal area .....                         | 27 |
| 3.1.5. Comparison of the different methods .....                      | 28 |
| 4. IT Framework.....                                                  | 29 |
| 4.1. ‘CNC Threading’ application .....                                | 29 |
| 4.2. Force Analysis .....                                             | 32 |
| 4.3. Automated Process.....                                           | 34 |
| 4.3.1. EMCO Compact 5 (Sinumerik 808d) .....                          | 34 |
| 4.3.2. AKIRA CNC Lathe Machine (Sinumerik 824d) .....                 | 36 |
| 4.3.3. Experiment.....                                                | 38 |
| 5. Conclusions .....                                                  | 40 |

|    |                                     |    |
|----|-------------------------------------|----|
| 6. | Short summary and ongoing work..... | 42 |
| 7. | References .....                    | 42 |
|    | APPENDIX.....                       | 45 |

# NOTATIONS

Latin letters

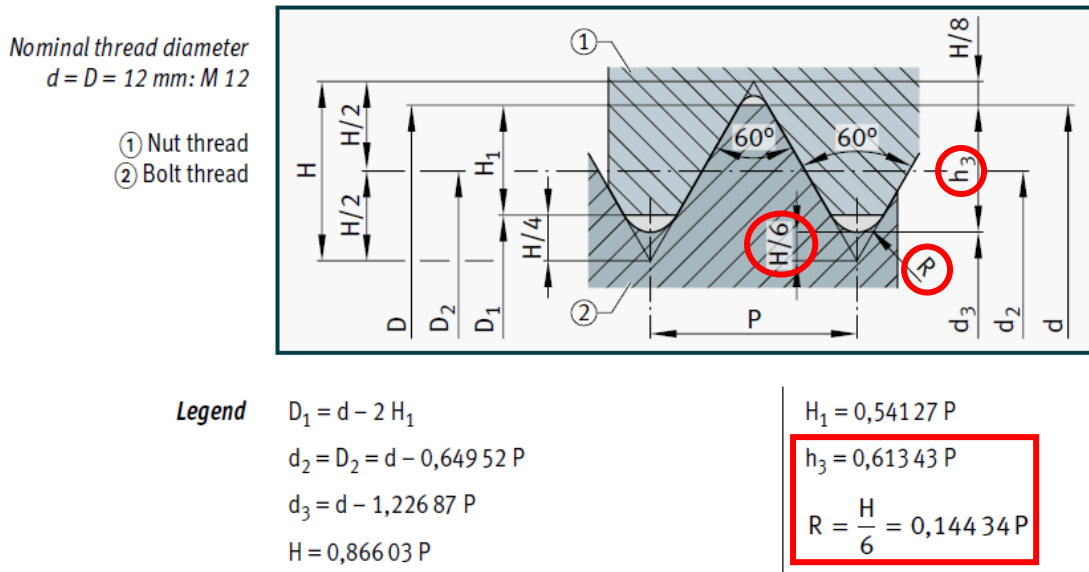
| Notation     | Name of the notation                           | Unit          |
|--------------|------------------------------------------------|---------------|
| $A_{Thread}$ | Total thread cross-sectional area              | $\text{mm}^2$ |
| $A_n$        | Thread cross-sectional area until cycle $n$    | $\text{mm}^2$ |
| $A_{xn}$     | Thread cross-sectional area only for cycle $n$ | $\text{mm}^2$ |
| $C$          | Cut Increment for EdgeCAM                      | —             |
| $N$          | Total number of passes                         | —             |
| $P$          | Thread Pitch                                   | mm            |
| $R$          | Threading root radius                          | mm            |
| $RE$         | Threading insert's nose radius                 | mm            |
| $a_p$        | Total thread depth                             | mm            |
| $a_{pn}$     | Thread depth at cycle $n$                      | mm            |
| $a_{pn_x}$   | Relative infeed depth for cycle $n$            | mm            |
| $d$          | Degression Factor for EdgeCAM                  | —             |
| $f_a$        | Feed rate                                      | mm/rev        |
| $h_n$        | Relative infeed depth for cycle $n$            | mm            |
| $n$          | Current pass of inspection                     | —             |
| $s$          | Spindle speed                                  | rev/min       |

# 1. INTRODUCTION

## 1.1.1. Problem

The convergence of mechanical engineering and information technology presents a dynamic challenge in optimizing threading operations on CNC lathe machines. This intricate task demands a profound grasp of both the mechanical intricacies governing insert geometry and threading processes, as well as the sophisticated IT infrastructure required for seamless operation optimization.

From a mechanical engineering standpoint, precision is paramount. It necessitates an in-depth study of existing solutions for thread creation, meticulous definition of insert geometry, and the orchestration of threading operations. This involves crafting a metric thread adhering to the ISO metric screw thread standard (ISO 9), ensuring a uniform load under machining. Given that the threading process unfolds in multiple cycles, each pass requires precise manipulation of dept of cut to reach equal load every time. Thanks to it, it can use the improved performance of the tool, avoid the brakeage, and decrease the machining time.



**Figure 1:** Cross-section of threaded connection.

Figure 1 represents a standard ISO metric thread connection between a bolt and a nut that is based on ISO 724, from which it can be determined that there are two main variables are needed for optimizing threading operation, which are the thread depth ( $h_3$ ) and the root radius ( $R$ ) and they depend on the thread pitch ( $P$ ):

$$h_3 = 0.61343 \cdot P, \quad (1)$$

$$R = 0.14434 \cdot P, \quad (2)$$

### **1.1.2. Aims and Overview**

This project mainly aims to optimize the performance during the threading operation and enhance the cutting insert lifetime. This can be done by decreasing the maximum forces during the machining process. This can be tested after automatically every machining process automatically. To reach this goal we have two main aspects:

- Mechanical Engineering Perspective
  - Study the existing solutions.
  - Threading insert's geometry.
    - Equation for the threading area.
- Information Technology 'IT' Framework
  - CNC communication for force measurement readings (On-going work).
  - Data analysis on force measurements.
  - Application for the calculations.
  - Make a closed loop algorithm for the communication (Future work)

### **Mechanical Engineering Perspective**

Within the realm of mechanical engineering, the focal point lies in meticulous examination and comparison of existing solutions. This involves a comprehensive study of various facets, commencing with software-driven built-in functions, exemplified by tools like EdgeCAM and SINUMERIK 808D. Additionally, a deep dive is undertaken into specialized catalogs from industry leaders such as Seco, Taegu-Tec, and Sandvik.

This project is driven by a comprehensive endeavor to conduct an in-depth analysis of EdgeCAM. The goal is to explore potential modifications to its input parameters. In parallel, for SINUMERIK 808D, the objective is to gain a comprehensive understanding of Cycle 99 for threading, with a view towards automating the G-code generation process. This generated G-code can then be seamlessly integrated into the SINUMERIK software for manual thread creation.

To attain the most optimal G-code, a profound comprehension of thread geometry is imperative. This knowledge enables the calculation of the total area, which is then evenly distributed across the requisite number of passes. While this approach does not explicitly account for the escalating vibrations associated with increased passes, it provides a robust approximation for maintaining consistent forces on the insert throughout the passes. This systematic method forms a cornerstone in the quest for enhanced threading operations on CNC lathe machines.

## **IT Framework**

The IT framework encompasses three pivotal aspects. The first facet involves crafting a robust communication protocol to seamlessly extract force measurements from the CNC machine. This initial step sets the stage for real-time data acquisition, an indispensable element in the decision-making process.

The second facet revolves around data analysis. This critical tool leverages the collected force measurements to discern patterns, determine average values, assess data distribution, and identify trends. These insights empower informed decisions concerning insert quality and operational efficiency.

The final pillar centers on the development of the 'CNC Threading' application. This innovative tool possesses the autonomy to generate G-Code instructions based on specific threading parameters, including pitch and diameter.

.

## 2. LITERATURE REVIEW

### 2.1.1. Industry 4.0

Industry 4.0, a monumental advancement in contemporary manufacturing, emerges from a continuum of industrial evolution spanning centuries. As seen in Figure 2, the first industrial revolution, characterized by mechanization, heralded a seismic transformation in production capacities. Subsequently, the second industrial revolution harnessed the potency of electrical energy, fundamentally altering manufacturing processes. The third industrial revolution, defined by pervasive digitalization, laid the foundation for the present-day paradigm of Industry 4.0 [1].

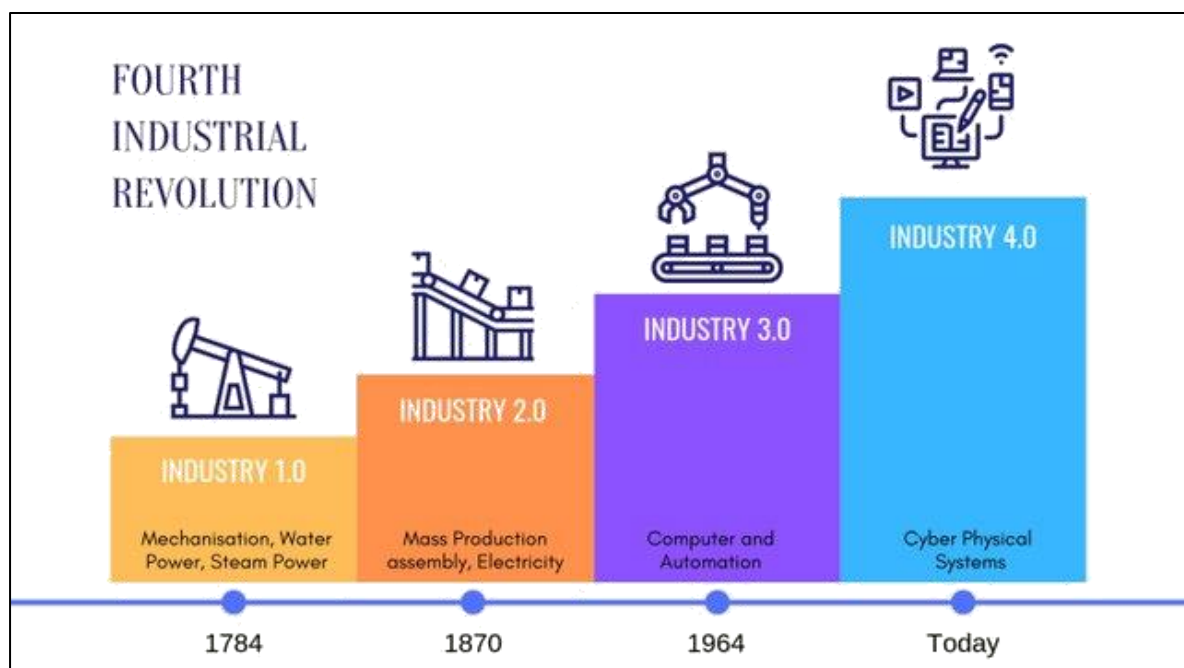


Figure 2: Industrial Revolution [1]

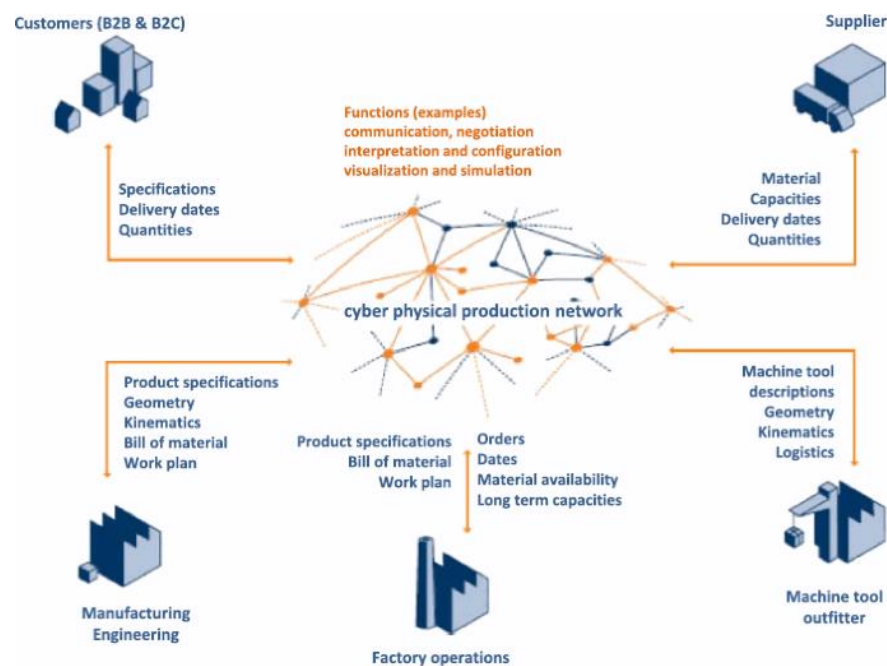
At its essence, Industry 4.0 embodies the convergence of digital technologies, the internet, and intelligent entities within factory ecosystems. This fusion facilitates seamless intercommunication between machinery, products and systems, heralding a new era of intelligent manufacturing. The integration of cyber-physical systems (CPS), the Internet of Things (IoT), and advanced analytics empowers factories to operate with unparalleled efficiency, adaptability and precision [2].

Within the realm of CNC machining, Industry 4.0 introduces a multitude of transformative possibilities. The infusion of intelligent technologies into CNC machines enables the acquisition and real-time analysis of data, facilitating dynamic adjustments to machining parameters.

Industry 4.0 is not merely an enhancement of technical capabilities; it paves the way for novel business models and value propositions. Practices of intelligent manufacturing enable just-in-time production, prognostic maintenance and an elevated level of customization, aligning production processes more closely with custom needs [3].

In essence, Industry 4.0 signifies a watershed moment in the annals of manufacturing, ushering in an era of unparalleled connectivity, intelligence and efficiency. By embracing the principles of Industry 4.0, it harnesses the potential of digitalization and smart technologies to augment the lifespan of cutting tools and optimize manufacturing operational effectiveness. As industries grapple with the demands of a fiercely competitive business landscape and the imperatives of mass customization and global outreach, the transition towards the transformative framework of Industry 4.0 gains rapid momentum.

This shift constitutes a manufacturing revolution, with the overarching aim of attaining maximal operational efficiency while minimizing resource expenditure [4]. Central to Industry 4.0 is the concept of "smart manufacturing" or "digital manufacturing," affording industries the capacity to execute adaptable production processes alongside personalized customization options [5].



**Figure 3:** Example for interdependencies of a supply chain in the context of the future project "Industry 4.0" [6]

Originating as a German initiative, Industry 4.0 envisions the creation of "smart factories" where manufacturing technologies undergo a profound metamorphosis through the integration of the Internet of Things (IoT) and cyber-physical systems (CPS) [2] as is represented in Figure 3. Within this paradigm, manufacturing systems possess the capability to monitor physical processes and generate digital counterparts, commonly referred to as "digital twins" of tangible

entities [5]. This advancement fosters real-time communication, intelligent decision-making, and seamless man-machine interaction within manufacturing organizations.

The integration of Industry 4.0 technologies has not only reconfigured production value chains but also revolutionized industry value chains and traditional business models of organizations. Key enabling technologies for Industry 4.0 practices encompass a spectrum of innovations, including the Internet of Things (IoT), cyber-physical systems (CPS), augmented and virtual reality, blockchain technology, machine learning (ML), artificial intelligence (AI), big data analytics and digital twins. It is anticipated that the adoption of Industry 4.0 technologies will not only bolster organizational efficiency but also contribute significantly to social and economic sustainability.

### **2.1.2. Different Communication Protocols for CNC**

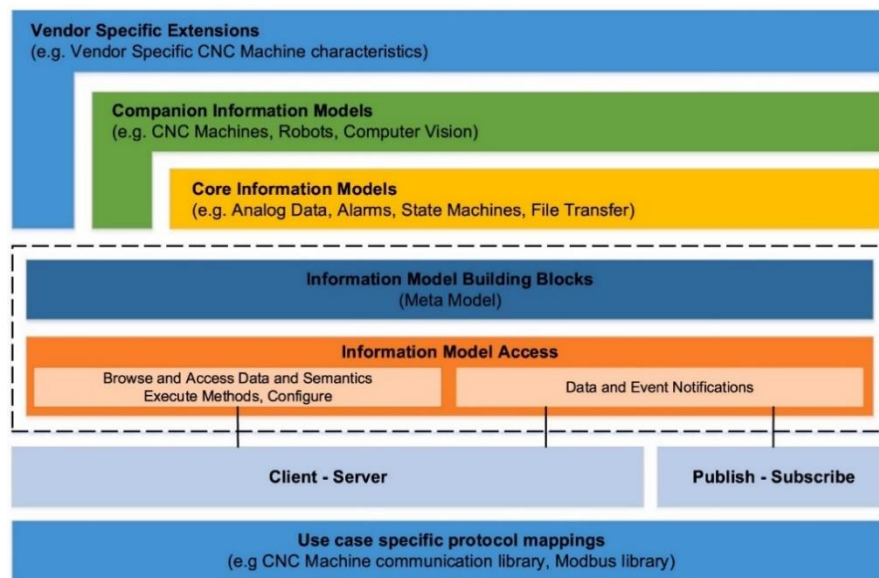
The integration of Industry 4.0 principles into CNC machining marks a significant leap forward in modern manufacturing. This paradigm shift harnesses advanced digital technologies to enhance CNC processes' efficiency, precision and adaptability. Crucial to this evolution are various communication protocols that facilitate seamless interaction between CNC machines and other components of the manufacturing ecosystem. In the following sections, we will delve into a detailed exploration of these protocols.

The OPC (Open Platform Communication) serves as the linchpin for industrial networking and telecommunication services. Originating in 1996 under the name OLE for Process Control (Object Linking and Embedding for Process Control), OPC has evolved into a cornerstone of interconnectivity within industrial processes [7]. Specifically designed to bridge the gap between Windows-based applications and process control devices, OPC standards offer a unified solution.

At the core of OPC architecture lies the concept of server-client communication. These standards were conceived to streamline efforts across manufacturers, ensuring a standardized instruction set across all OPC servers and clients, regardless of their respective manufacturers. This uniformity presents a shared interface for other industry players, such as SCADA (Supervisory Control And Data Acquisition) or HMI (Human Machine Interface) producers, fostering seamless integration within diverse industrial ecosystems.

Introduced in 2008, the OPC Unified Architecture (OPC UA) represents a watershed moment in industrial communication [8]. Its platform-independent, service-oriented architecture seamlessly integrates all functionalities of the individual OPC Classic specifications into a cohesive and extensible framework. This unified approach ensures functional equivalence by meticulously mapping all COM OPC Classic specifications to UA.

Furthermore, UA's platform independence spans from embedded micro-controllers to cloud-based infrastructures, offering unparalleled versatility. Security is paramount, with UA implementing robust measures such as encryption, authentication, and auditing. Its extensibility allows for the seamless integration of new features without disrupting existing applications. In addition, UA's comprehensive information modeling capabilities enable the precise definition of intricate data structures, underpinning a sophisticated understanding of information within the system. This holistic approach within OPC UA addresses the critical imperatives of interoperability, security and adaptability in contemporary industrial communication systems.



**Figure 4:** OPC UA architecture. [7]

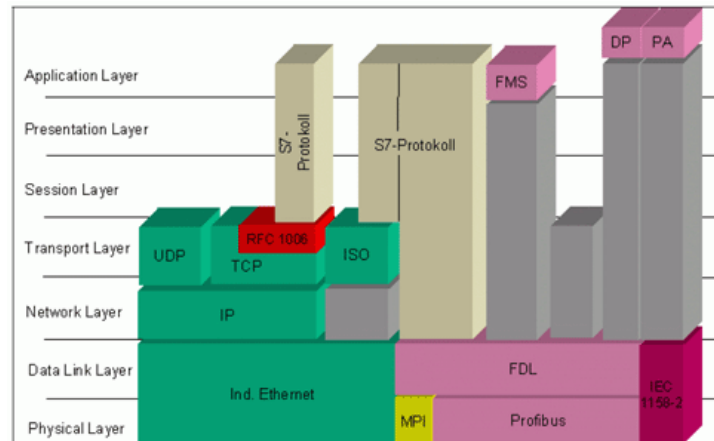
Ethernet, a bedrock of modern networking, plays a pivotal role in CNC communication. Renowned for its high data transmission rates and reliability, Ethernet provides a robust platform for real-time data exchange between CNC machines and control systems. Through protocols like Modbus TCP/IP and Ethernet/IP, Ethernet facilitates seamless integration, enabling CNC systems to operate in concert with other components of the manufacturing process [8, 9].

The S7 Protocol, specifically tailored for environments utilizing Siemens PLCs, plays a foundational role in CNC operations. It is meticulously designed to facilitate seamless communication between Siemens devices and systems, thereby enabling cohesive coordination and data exchange between CNC machines and Siemens PLCs. This integration significantly enhances workflow synchronization and augments the automation capabilities of CNC processes [10].

The S7 Protocol (RFC 1006) stands as a pivotal enabler for establishing seamless connections between S7 automation devices and a diverse array of communication partners. It

provides a direct conduit to the S7 user memory, all the while obviating the need for alterations in the user application itself [11].

This protocol, as illustrated in the ISO - OSI classification chart (see Figure 5) [11], offers robust support for a range of transportation methods. When opting for TCP/IP, a communication unit (CP) designed for Ethernet connection, or an onboard Ethernet interface that supports ISO-on-TCP (RFC1006), becomes essential.



**Figure 5:** Chart ISO - OSI classification [11].

The S7 Protocol (RFC 1006) sets itself apart from Simatic Fetch/Write by affording the ability to address all internal components on the PLC side, devoid of constraints on data blocks (DB). Depending on the specific PLC, minimal or even no configuration adjustments may be necessary to seamlessly integrate the S7 Protocol.

Furthermore, the SIMATIC S7 series natively supports the ISO-on-TCP protocol in passive mode. In this capacity, it functions as a server, readily accepting client connections. This innate feature enhances the adaptability and reliability of data exchange within the manufacturing ecosystem. Additionally, this protocol adheres to ISO - OSI classification standards, solidifying its compatibility within the broader framework of Industry 4.0 and CNC machining operations.

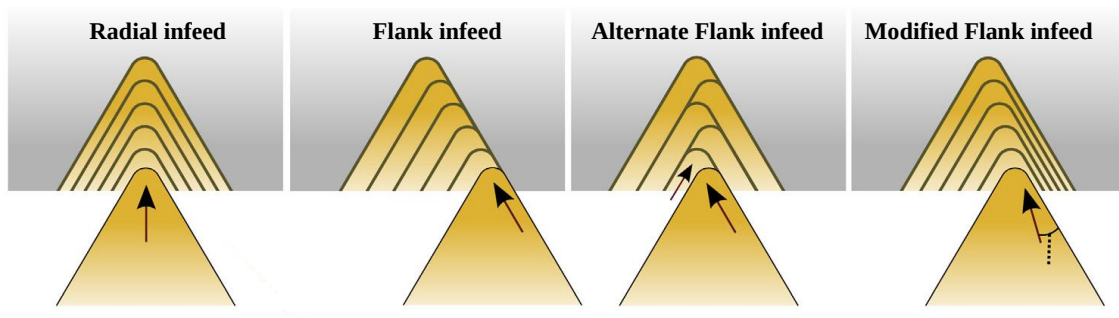
In summary, the convergence of Industry 4.0 and CNC machining represents a transformative juncture in modern manufacturing. The adoption of communication protocols like OPC, Ethernet, and S7 Protocol facilitates seamless interaction, enabling CNC machines to operate intelligently within the broader manufacturing ecosystem. These protocols not only enhance operational efficiency but also fortify the security and coordination of CNC operations.

### 2.1.3. Threading Operations

In the realm of thread turning operations, the longevity and performance of cutting inserts stand as pivotal factors in ensuring sustained machining efficiency. The operational lifespan of an insert is inherently tied to the forces it contends with during the cutting process. Maintaining

these forces at a consistent level is a paramount objective, as it not only prolongs the insert's longevity, but also mitigates the maximum stresses it endures. Achieving this constancy in forces is central to enhancing the insert's operational efficiency and overall effectiveness in machining processes. This focus on force control holds the promise of significantly extending the lifespan of cutting inserts, thereby contributing to enhanced productivity and cost-effectiveness in industrial applications. However, unlike conventional turning, thread turning introduces unique challenges. The axial feed rate  $f_a$  and spindle speed  $s$  are intrinsically linked, necessitating the tool to traverse one pitch  $P$  per spindle revolution [12]. This distinctive characteristic, coupled with form cutting, entails wide chips and substantial cutting forces.

Moreover, due to machining limitations, the entire thread depth cannot be achieved in a single pass. Instead, the cutting insert gradually approaches the root over several passes from range 6 to 12 passes as represented in Figure 6 [12]. Various infeed strategies govern this progression, influencing chip evacuation, surface quality, and the distribution of cutting forces. For example, radial infeed minimizes axial forces for symmetrical inserts, while flank infeed ensures better chip flow despite potential surface quality trade-offs [12].



**Figure 6:** Different types of infeed strategy [13].

While extensive research has been devoted to modeling forces in various form cutting operations, such as hobbling and gear cutting[14]–[16], theoretical investigations specific to threading operations remain relatively scarce. Notable studies by Fromentin and Poulachon [17], [18], Araujo et al.[19], and Jun and Araujo [20] have presented force models for thread milling, which, while insightful, do not directly apply to thread turning. The intricacies of multi-point thread turning, characterized by arbitrary insert profiles and infeed plans, render the utilization of existing force prediction models impractical [21]. Akyildiz and Livatyali's preliminary force calculation method is limited in scope and not suitable for arbitrary insert profiles and infeed plans [22].

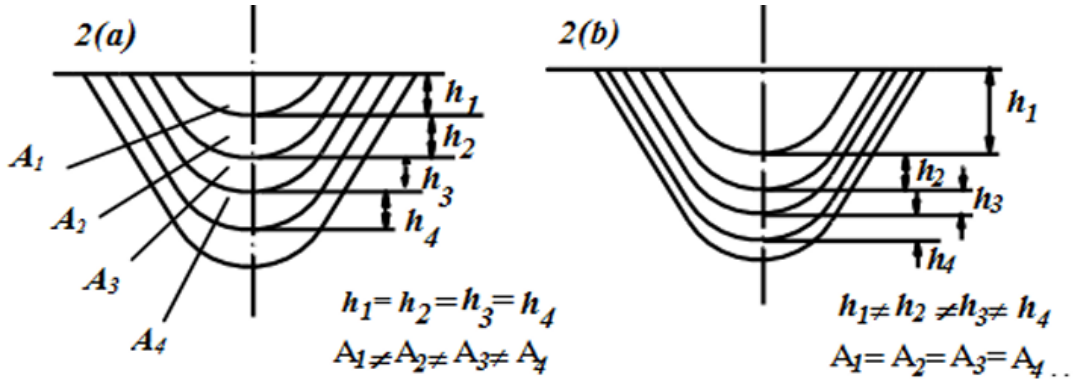


**Figure 7:** Cross-sectional geometry of thread chips: **a.** First pass, **b.** Intermediate pass, **c.** Last pass [23]

Another study by Hamza Akyildiz [23] focuses on predicting cutting forces during thread cutting using a mechanistic force model, in which the cutting depth is kept constant on every cycle that result in chip formation as shown in Figure 7, where we can see that the machined cross-section is increasing in every cycle.

### Existing solutions for thread operations

Research by F. Sineider Martins, R. Reina-Muñoz, and V. Martins Lira [24], the presented a data acquisition system for measuring the cutting force in the threading process on conventional lathes, utilizing analytical models to determine the depth of cut of the machining tool trying to achieve the most stable forces on the insert based on a constant are in each cycle as shown in Figure 8 (b).



**Figure 8:** Threading Strategies for radial infeed: **(a)** constant **(b)** variable. [24]

The authors analyze the, Sandvik [25], Lira [26], and Fredericks [27] models to draw conclusions about the cutting force from each model on thread pitch  $P = 1.5$ . The formulas of these models are presented in Equations 3 - 5:

Sandvik's model: 
$$a_{pn} = \frac{a_p}{\sqrt{N-1}} \cdot \sqrt{n-1} \text{ except for } a_{p1} = \frac{a_p}{\sqrt{N-1}} \cdot \sqrt{0.3}, \quad (3)$$

Lira's model: 
$$a_{pn} = \sqrt{\frac{5}{4} \cdot R^2 + \frac{n}{N} \left[ a_p^2 \cdot 2 \cdot R \cdot a_p - \frac{1}{4} \cdot R^2 \right]} - R, \quad (4)$$

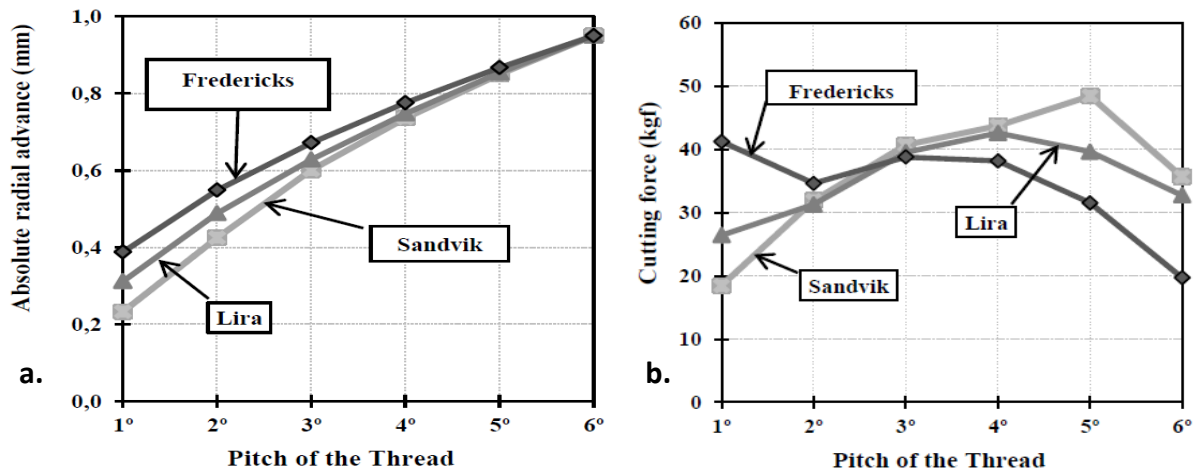
Frederick's model: 
$$a_{pn} = a_p \cdot \sqrt{\frac{n}{N}}, \quad (5)$$

In which,  $a_{pn}$  denotes the total depth in cycle  $n$ ,  $N$  signifies the overall number of passes. While  $a_p$  represents the total thread depth that is based on Equation 1. Even though this value is standardized but there are tolerances affecting the final machining value, so the thread depth  $a_p$  can be calculated based on Equation 6, where  $es$  and  $T_{d_2}$  are based on the selected basic deviation and tolerance grade.

$$a_p = 0.61343 * P + \frac{es}{2} + \frac{T_{d_2}}{4}, \quad (6)$$

Also,  $R$  is the thread root radius that is based on Equation 2 from the standard ISO 965, but it depends on the manufacturer of the insert that they tend to modify this value slightly.

The experiments used three proof samples configured for the study, and the data obtained from the models show that the Fredericks algorithm presents the greatest initial radial advance, while the Sandvik model shows the smallest incremental radial advance variation and this is based on Figure 9 (a).



**Figure 9: a. Absolute radial advance for each model b. Average cutting force for each model.**

According to that, Frederick's model had the highest force in the first cycle, but this force is less than the maximum force obtained by Sandvik's model. And this can be seen in Figure 9 (b) that shows that among the evaluated models, Sandvik's exhibited a progressive force increase until the fifth pass, followed by a slight drop in the sixth. This pattern is attributed to nearing thread completion in the fifth pass. However, it displayed a notable force increment between passes, potentially leading to tool failure. Lira's model also saw a progressive force rise, up to the fourth pass, after which force diminished. This decrease was due to the nearing thread completion. Interestingly, it showed the least force increment between passes. Fredericks' model demonstrated a unique behavior, featuring a continuous force reduction across passes. Notably, it showed the lowest maximum and average cutting forces, suggesting potential for reduced tool wear. Among the models, Fredericks' stands out for its potential to minimize tool wear, while Lira's exhibits the least force variation between passes.

a. EdgeCAM

EdgeCAM stands as a sophisticated software solution that wields significant influence over threading operations. At its core lies a crucial parameter known as the degression factor  $d$ , which is a factor to calculate how the depth of cut for each successive pass is reduced, and it is the main parameter that affects the radial infeed depth of the threading operation as represented in Equation 7.

$$a_{pn} = C \cdot \sqrt[d]{n} , \quad (7)$$

Where  $C$  is the Cut Increment and can be obtained as  $C = \frac{a_p}{\sqrt[d]{N}}$

The suggested value for the  $d$  is 2 that aims to produce a constant volume of cut in every pass, while if it is 1, it produces a constant depth of cut, this parameter needs to be studied, and optimized if needed.

b. SINUMERIK 808D [Cycle 99]

The SINUMERIK 808D system features an exclusive function dedicated to thread depth determination. However, the precise algorithm underlying this function remains proprietary and closely guarded by the manufacturer. This sophisticated tool is designed to generate precise threads based on specific pitch parameters. Serving as a pivotal element, this proprietary function forms the foundation for deriving the infeed depths. A systematic processing of this function for individual thread pitches yields the necessary depth values, which are then systematically cataloged in a database for further analytical purposes. This strategic approach adeptly circumvents the challenge presented by the undisclosed equation.

c. Catalogues

Other methods include the manually generated G-codes based on suggested values for the radial infeed depth by the Catalogues, such as Seco catalogue in Figure 10. It also does not provide the equation to calculate the depth, but it provides the database with the infeed depth of each cycle, that can be used to generate the G-code.

## Thread turning – Cutting data



### Number of passes and infeed depths

#### External ISO-metric threads

| Ph    | 6.0  | 5.5  | 5.0  | 4.5  | 4.0  | 3.5  | 3.0  | 2.5  | 2.0  | 1.75 | 1.5  | 1.25 | 1.0  | 0.80 | 0.75 | 0.50 |
|-------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| $a_p$ | 3.82 | 3.52 | 3.19 | 2.87 | 2.53 | 2.23 | 1.92 | 1.60 | 1.25 | 1.13 | 0.93 | 0.81 | 0.65 | 0.52 | 0.48 | 0.33 |
| 1     | 0.46 | 0.43 | 0.41 | 0.37 | 0.34 | 0.34 | 0.28 | 0.27 | 0.24 | 0.22 | 0.22 | 0.21 | 0.18 | 0.17 | 0.16 | 0.11 |
| 2     | 0.43 | 0.40 | 0.39 | 0.34 | 0.32 | 0.31 | 0.26 | 0.24 | 0.22 | 0.20 | 0.20 | 0.17 | 0.16 | 0.15 | 0.14 | 0.09 |
| 3     | 0.35 | 0.32 | 0.32 | 0.28 | 0.25 | 0.25 | 0.21 | 0.20 | 0.18 | 0.17 | 0.17 | 0.14 | 0.12 | 0.12 | 0.11 | 0.07 |
| 4     | 0.30 | 0.28 | 0.27 | 0.24 | 0.22 | 0.21 | 0.18 | 0.17 | 0.16 | 0.14 | 0.14 | 0.11 | 0.11 | 0.08 | 0.07 | 0.06 |
| 5     | 0.29 | 0.26 | 0.24 | 0.22 | 0.20 | 0.18 | 0.16 | 0.15 | 0.14 | 0.12 | 0.12 | 0.10 | 0.08 | –    | –    | –    |
| 6     | 0.26 | 0.24 | 0.24 | 0.22 | 0.18 | 0.18 | 0.15 | 0.15 | 0.12 | 0.10 | 0.08 | 0.08 | –    | –    | –    | –    |
| 7     | 0.24 | 0.21 | 0.22 | 0.20 | 0.17 | 0.16 | 0.14 | 0.12 | 0.11 | 0.10 | –    | –    | –    | –    | –    | –    |
| 8     | 0.23 | 0.20 | 0.20 | 0.18 | 0.15 | 0.15 | 0.13 | 0.11 | 0.08 | 0.08 | –    | –    | –    | –    | –    | –    |
| 9     | 0.22 | 0.19 | 0.19 | 0.17 | 0.14 | 0.14 | 0.12 | 0.11 | –    | –    | –    | –    | –    | –    | –    | –    |
| 10    | 0.19 | 0.18 | 0.18 | 0.16 | 0.13 | 0.12 | 0.11 | 0.08 | –    | –    | –    | –    | –    | –    | –    | –    |
| 11    | 0.18 | 0.17 | 0.16 | 0.14 | 0.12 | 0.11 | 0.10 | –    | –    | –    | –    | –    | –    | –    | –    | –    |
| 12    | 0.16 | 0.15 | 0.15 | 0.13 | 0.12 | 0.08 | 0.08 | –    | –    | –    | –    | –    | –    | –    | –    | –    |
| 13    | 0.15 | 0.14 | 0.12 | 0.12 | 0.11 | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    |
| 14    | 0.13 | 0.13 | 0.10 | 0.10 | 0.08 | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    |
| 15    | 0.13 | 0.12 | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    |
| 16    | 0.10 | 0.10 | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    | –    |

**Figure 10:** Number of passes and infeed depths of external ISO-metric threads. [28]

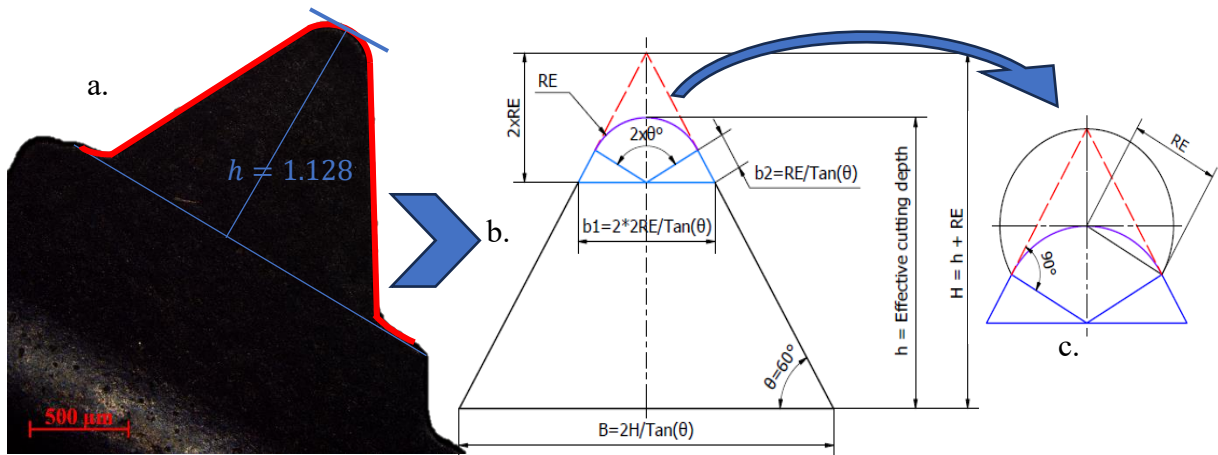
### 3. Mechanical Engineering Perspective

Within the domain of mechanical engineering, this study delves into an exhaustive examination and comparison of various solutions. The focus extends from software-driven built-in functions, exemplified by tools like EdgeCAM and SINUMERIK 808D, to specialized catalogs from industry leaders such as Seco, and Sandvik catalogues. To do this study mainly the thread cross-section needs to be analyzed and accordingly derive a new equation that shall produce the equal volume in every pass.

#### 3.1.1. Calculation of the thread area

As the initial phase in optimizing the threading process, a precise definition of the thread's cross-sectional area is imperative, ensuring equal division in every cycle. This optimization begins with the analysis of a microscopic image of a 16ER1.75ISO insert from Seco catalogue, as depicted in Figure 11 (a). Subsequently, Figure 11 (b) portrays the conceptual cross-section of the entire thread, deliberately excluding the consideration of side fillet radii for the sake of simplicity, given their negligible impact at this stage.

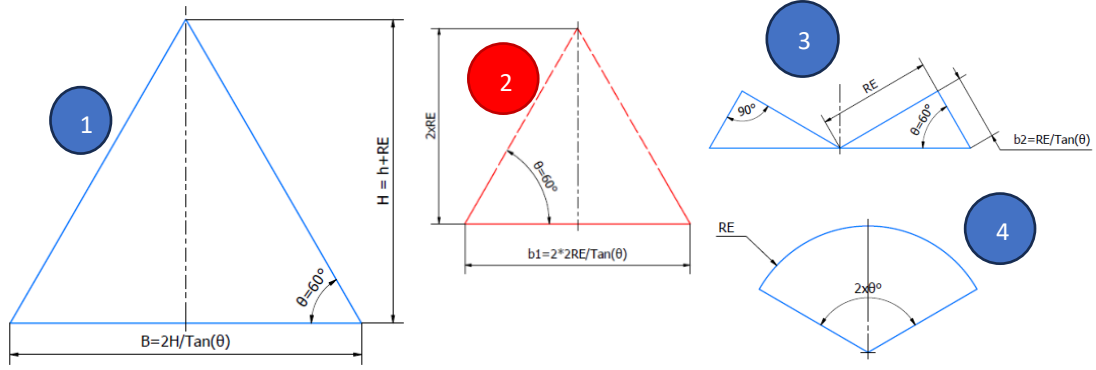
Adhering to the ISO metric thread standard showcased in Figure 1, the distance from the imaginary sharp edge to the thread root radius is denoted as  $H/6$ , with the standard radius  $RE$  equivalently being  $H/6$ . Consequently, the computation of the distance from the sharp edge to the center of the nose radius, representing the height of the smaller triangle, results in  $2 \cdot RE$ , as visually illustrated in Figure 11.



**Figure 11:** a. microscopic image of a 16ER1.75ISO insert from Seco b. General thread cross-section. c. Threading tip with imaginary Thales's Theorem

Also based on Thales's theorem, if we draw a circle centered at the top of the  $RE$  circle and draw a line to the 90° edge as shown in Figure 11 (c), this line should pass through all the edges of the triangle and so, the distance will be also  $RE$ .

The total cross-section of the thread represented in Figure 11 (b) can be divided into four geometrical parts to evaluate the total area, those can be seen in Figure 12.



**Figure 12:** General thread partitions.

1. The large triangle with base B:
  - $A_1 = \frac{(h + RE)^2}{\tan \theta}$
2. The tip which is a smaller triangle with base b1:
  - $A_2 = \frac{4 \cdot RE^2}{\tan \theta}$
  - We will need to subtract this area.
3. The side triangles which are symmetric:
  - $A_3 = \frac{RE^2}{\tan \theta}$
4. The circular sector:
  - $A_4 = \frac{2 \cdot \theta^\circ}{360^\circ} \cdot \pi \cdot RE^2$

The overall thread cross-sectional area  $A_{Thread}$  is determined by the following formula:

$$\circ A_{Thread} = A_1 - A_2 + A_3 + A_4$$

This expression can be further refined as:

$$A_{Thread} = \frac{(h+RE)^2 - RE^2(3 - \frac{\pi}{3} \tan \theta)}{\tan \theta}, \quad (8)$$

To calculate the total area of the specific thread, the cutting depth is considered as the total thread depth  $h = a_p$  that utilizes Equation 6, so the area can be calculated at any determined depth. Simultaneously, the absolute infeed depth at  $n^{th}$  cycle can be derived as in Equation 9.

$$a_{pn} = \sqrt{\left(A_n \cdot \tan \theta + RE^2 \cdot \left(3 - \frac{\pi}{3} \cdot \tan \theta\right)\right)} - RE \quad (9)$$

In which  $A_n$  is the area at cycle  $n$  and can be calculated as  $A_n = \frac{A_{Thread}}{N} \cdot n$  that assures a constant volume in every cycle, the  $RE$  is the nose radius of the threading insert that is depends on the manufacturer, but it has to be close to the thread root radius  $R$  from Equation 2.

The calculation of thread area hinges on two primary parameters: the total depth of the thread, based on the pitch and tolerance grade of the generated thread, and the thread radius, while to derive the infeed depth, these parameters are augmented by the number of passes.

### 3.1.2. Thread Area of different methods

Based on Equations 8 and 9, the absolute infeed depth and the area at any specific infeed can be calculated, and accordingly a comparison between the different threading methods can be established to be able to enhance the overall threading operation. To demonstrate this, two different metric threads will be tested to clarify the size effect, which are M12x1.75 and M6x1.0 that have dimensions as mentioned in Table 1.

**Table 1:** M12x1.75 & M6x1 threads specifications. [28] [29]

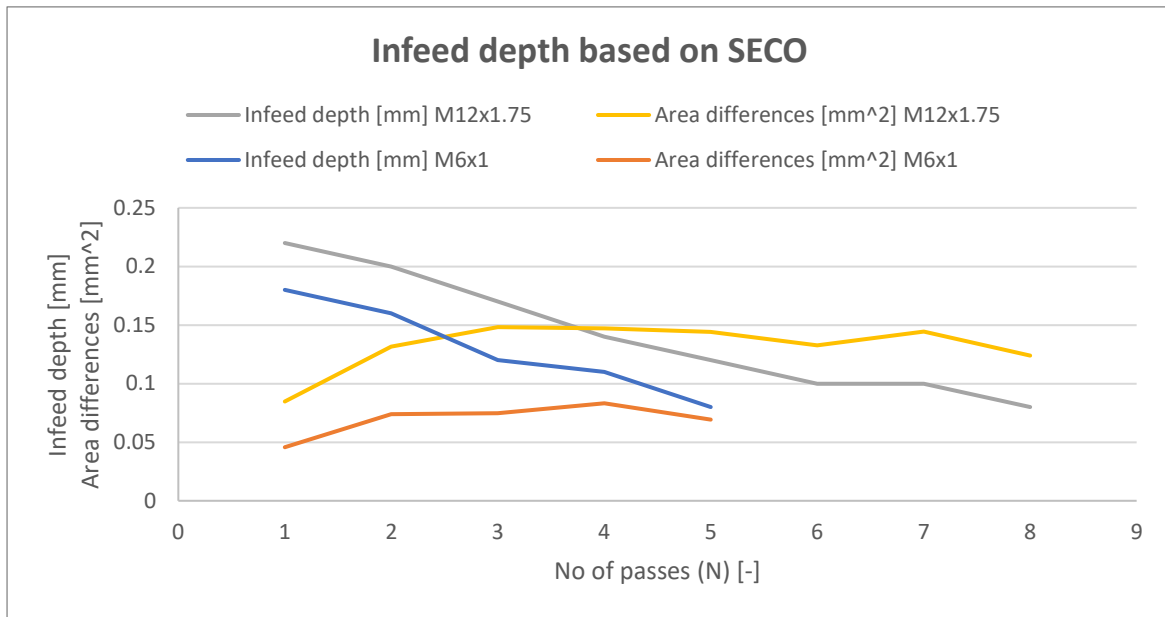
|                                  |                    | <b>M12x1.75</b> | <b>M6x1.0</b> |
|----------------------------------|--------------------|-----------------|---------------|
| <b><i>P</i></b>                  | [mm]               | 1.75            | 1.0           |
| <b><i>RE</i></b>                 | [mm]               | 0.25            | 0.14          |
| <b><i>es</i></b>                 | [mm]               | 0.034           | 0.026         |
| <b><i>N</i></b>                  | [—]                | 8               | 5             |
| <b><i>T<sub>d2</sub></i></b>     | [mm]               | 0.15            | 0.132         |
| <b><i>h = a<sub>p</sub></i></b>  | [mm]               | 1.128           | 0.654         |
| <b><i>A<sub>Thread</sub></i></b> | [mm <sup>2</sup> ] | 1.053           | 0.351         |

#### a. Seco Catalogue

Since we are studying the inserts made by Seco, we will have to apply the values of the infeed depth suggested in their catalogue that was mentioned in Figure 10 and based on these values we calculate the area of each cycle based on Equation 8 that can be utilized to get the area for cycle  $n$  as  $A_n = \frac{(a_{pn} + RE)^2 - RE^2 (3 - \frac{\pi}{3} \tan 60^\circ)}{\tan 60^\circ}$ , where  $a_{pn}$  is the absolute depth at  $n^{th}$  pass.

**Table 2:** Infeed depth for **M12x1.75** and **M6x1.0** based on Seco catalogue.

|                 | <b><i>P = 1.0</i></b>       |                              | <b><i>P = 1.75</i></b>      |                              |
|-----------------|-----------------------------|------------------------------|-----------------------------|------------------------------|
| <b><i>n</i></b> | <b><i>h<sub>n</sub></i></b> | <b><i>A<sub>xn</sub></i></b> | <b><i>h<sub>n</sub></i></b> | <b><i>A<sub>xn</sub></i></b> |
| <b>1</b>        | 0.18                        | 0.046                        | 0.22                        | 0.085                        |
| <b>2</b>        | 0.16                        | 0.074                        | 0.20                        | 0.132                        |
| <b>3</b>        | 0.12                        | 0.075                        | 0.17                        | 0.148                        |
| <b>4</b>        | 0.11                        | 0.083                        | 0.14                        | 0.147                        |
| <b>5</b>        | 0.08                        | 0.069                        | 0.12                        | 0.144                        |
| <b>6</b>        | —                           | —                            | 0.10                        | 0.133                        |
| <b>7</b>        | —                           | —                            | 0.10                        | 0.144                        |
| <b>8</b>        | —                           | —                            | 0.08                        | 0.124                        |

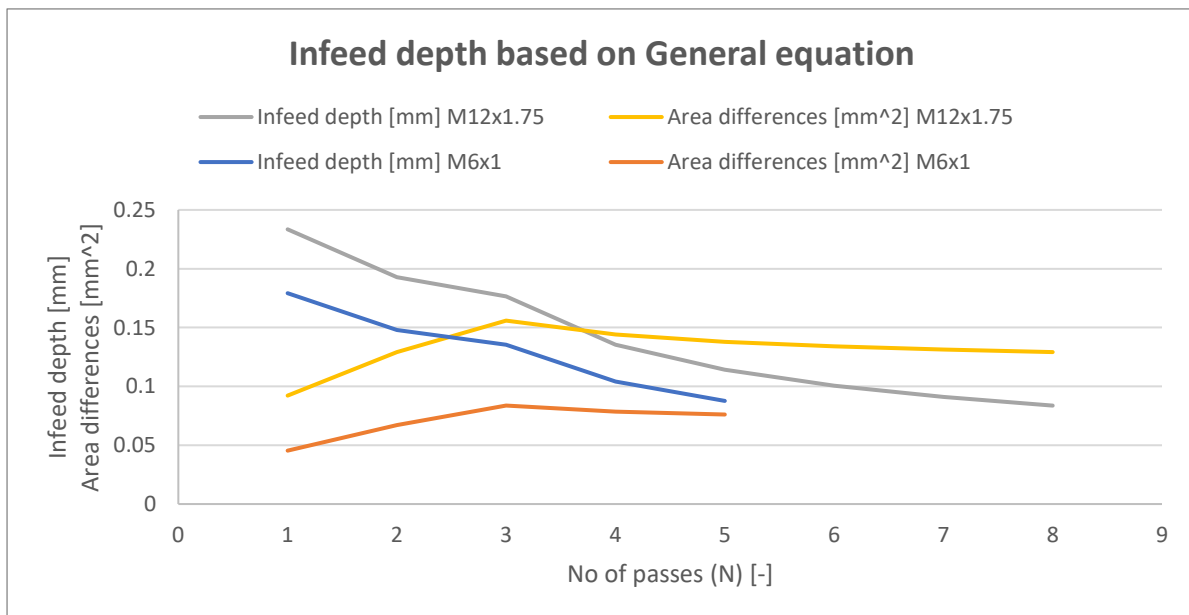


**Figure 13:** Infeed depth based on SECO catalogue.

From the graphical representation in Figure 13, reveals consistent fluctuations in the thread area across cycles, with each cycle undergoing changes within a defined margin. The initial pass consistently exhibits a smaller area than the subsequent cycles, possibly as a precautionary measure to accommodate variations in the material of the threaded cylindrical surface. This pattern is observed in both cases. Seco specifically provides values for coarse threads with a tolerance of 6g.

b. Sandvik's model [General Equation]

The general equation is the same method that was used by Sandvik in its model represented in Equation 3, and the depth is used to calculate the area based on Equation 8.

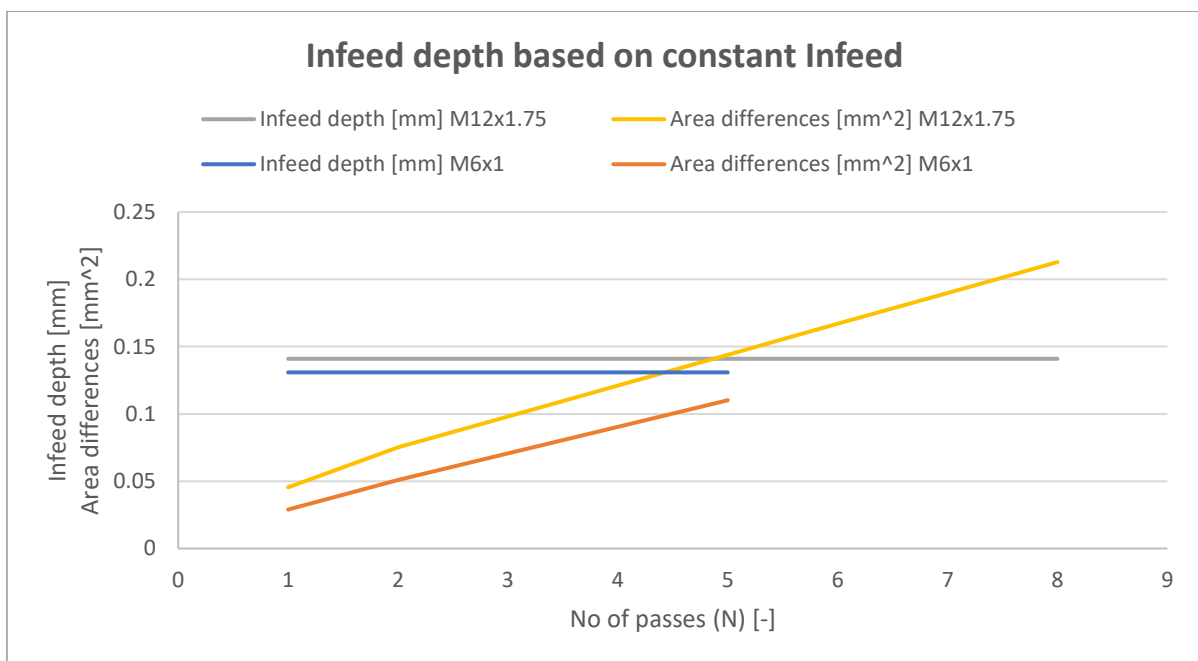


**Figure 14:** Infeed depth based on General equation.

The graph in Figure 14 illustrates a smaller area for the initial two passes, followed by a trend towards consistent values for subsequent passes. While the first pass avoids overloading the insert similar to the Seco case, the ensuing passes exhibit more significant fluctuations. In both threads, certain passes incur a larger area, mirroring the pattern observed in case of the Seco catalogue method and resulting in increased insert load during specific passes.

c. Constant infeed

A straightforward yet imprecise approach involves dividing the total thread depth by the number of passes, yielding a consistent infeed depth at each stage. However, this method leads to a significant increase in the machining area with each step, as depicted in Figure 15, which means higher forces are generated on the insert in each cycle, that may lead to an immediate break of the insert or it would decrease the lifetime of it dramatically.



**Figure 15:** Infeed depth based on constant Infeed.

d. SINUMERIK 808D

To retrieve values from SINUMERIK, Cycle 99 is employed to generate the thread path using variation 300103. The actual position of each pass is then subtracted to derive the infeed depth, as illustrated in Figure 16.

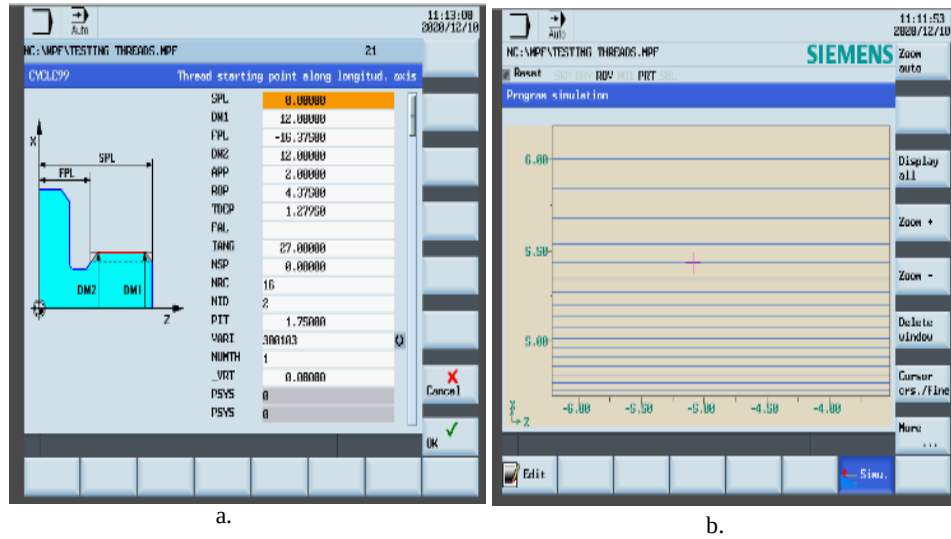


Figure 16: a. SINUMERIK cycle99 parameters. b. Tool path.

Through multiple trials on the SINUMERIK software, it has been established that the infeed depth is solely determined by the pitch and remains unaffected by the nominal diameter of the thread, also it was clear the similarity with the general equation results. Figure 17 presents a comparison between the area of SINUMERIK and the general equation models.

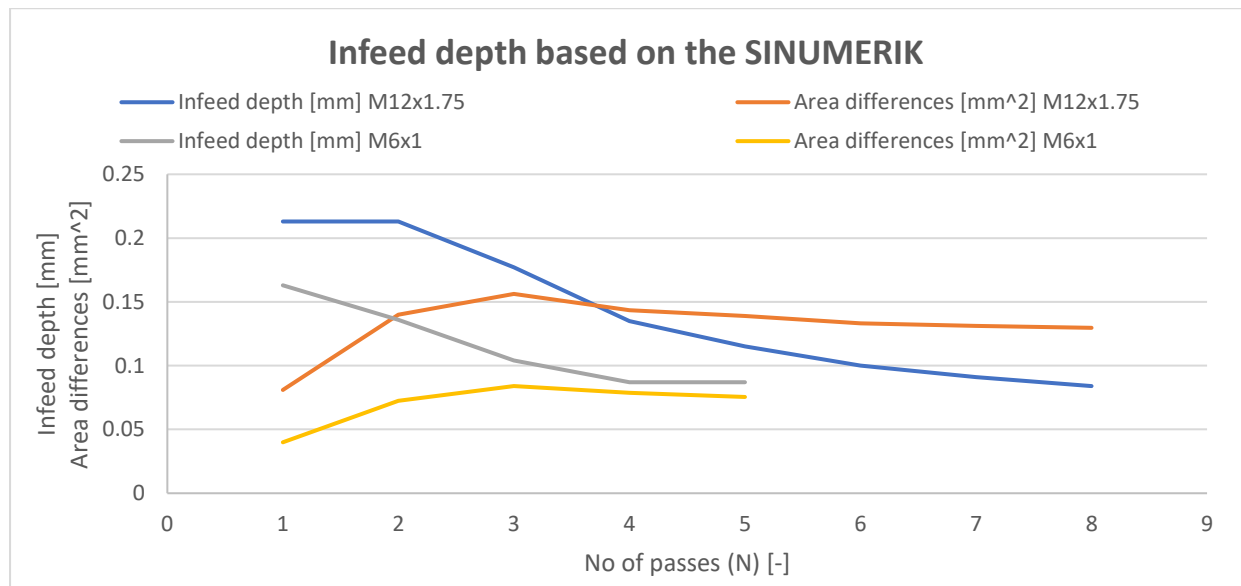


Figure 17: Infeed depth based on the SINUMERIK compared to the General Equation.

The graphs from Figure 17 and Figure 14 illustrates a remarkable proximity between the area differences based on SINUMERIK and the values derived from the general equation, suggesting a potential correlation between the two.

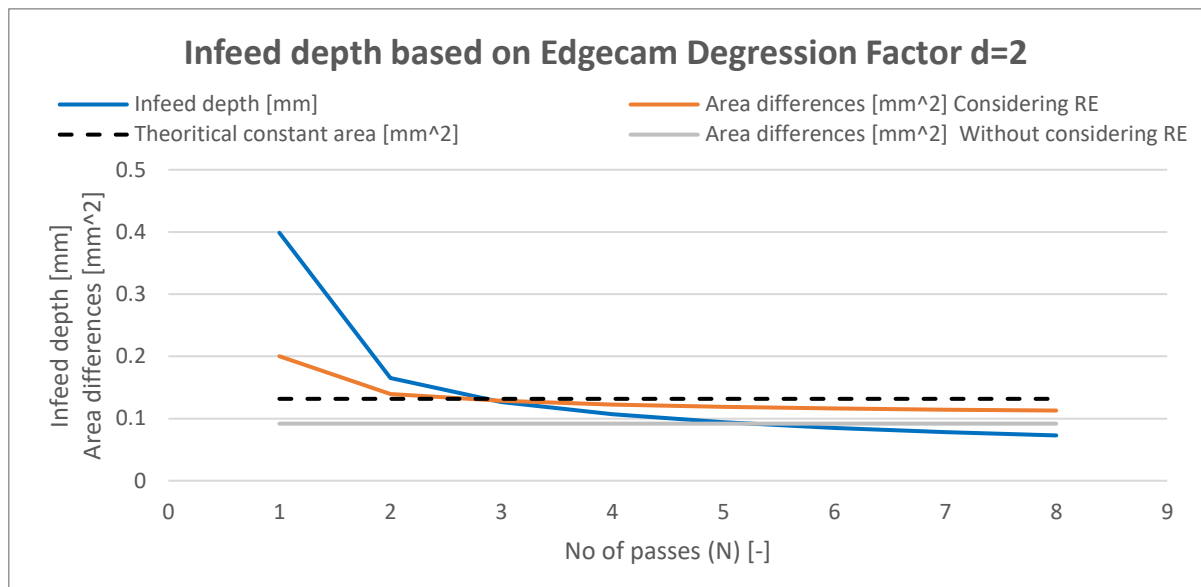
#### e. EdgeCAM

In Section 0 the EdgeCAM method was presented, utilizing the cutting depth calculation given by Equation 7, that can be simplified as  $a_{pn} = a_p \cdot \sqrt[d]{n/N}$ , where  $d$  is the degression

factor. Initial examination involves checking the behavior of the suggested degression factor for the case of M12x175, with relevant parameter listed in Table 3.

**Table 3:** EdgeCAM parameters for infeed depth calculations.

|                            |       |
|----------------------------|-------|
| Cutting depth:             | 1.128 |
| Degression Factor ( $d$ ): | 2     |
| Number of passes ( $N$ ):  | 8     |

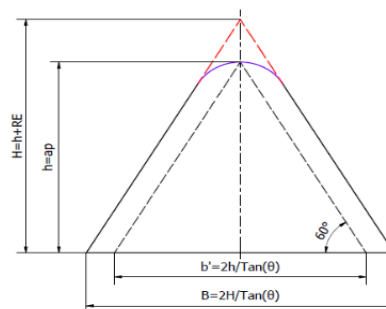


**Figure 18:** Degression factor for  $d=2$  for M12x1.75.

The graphical representation of Figure 18 shows that when we considered the nose radius of the insert which is the real geometry of the thread, it doesn't give a constant function rather it shows a much higher force in the first cycle.

The area of the thread without considering the nose radius decreased, and this is because we consider the same depth but in this case the insert has a sharp edge not rounded, this means that the EdgeCAM considers smaller area as shown in Figure 19 as the dashed black lines.

As a conclusion, Edgecam doesn't consider the real geometry of the insert and it just considers the triangle cross section of the thread, which means that the area removed based on the EdgeCAM considerations, is less than the real area removed during machining.

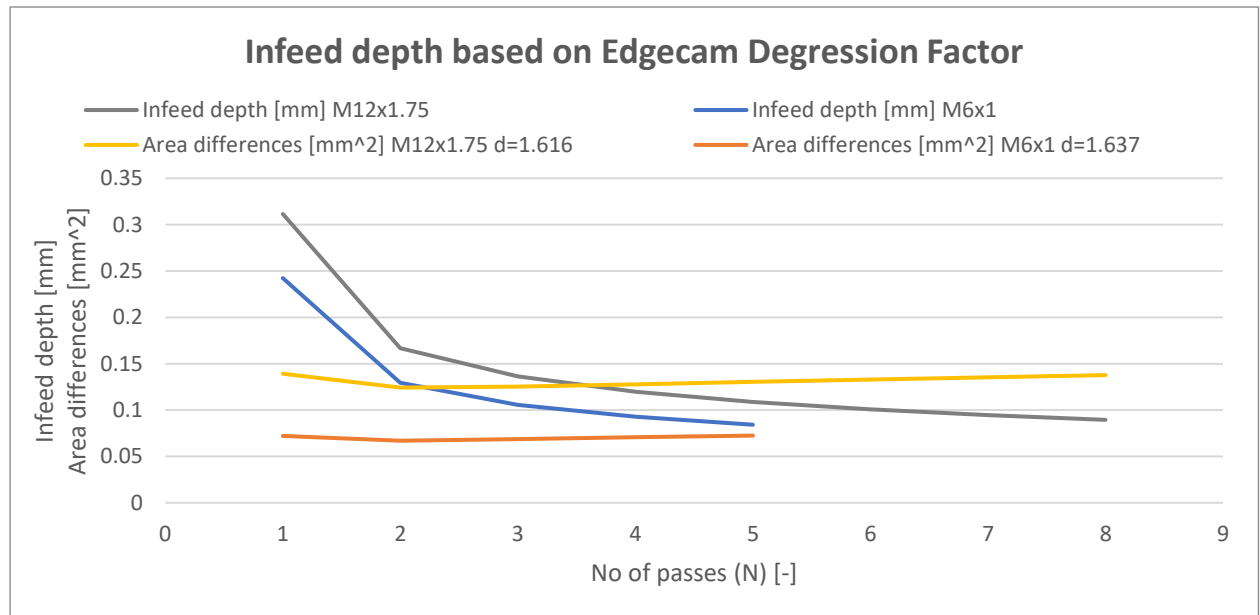


**Figure 19:** Insert shape Considered by EdgeCAM.

In this case, we need to find another value for the degression factor that gives the closest function for the constant area. To do that we will calculate square difference of regression value and find the degression factor that gives us the smallest value of  $R^2$ .

$$\sum R^2 = \sum_{n=1}^{n=N} \left( A_n - \frac{A_{Thread}}{N} \right)^2, \quad (10)$$

Using those functions we find the optimal degression factor for M12x1.75 is  $d = 1.616$ , while for M6x1.0 is  $d = 1.637$ . The graph in Figure 20 shows that by choosing the correct value for the degression factor it gives us a curve which is very close to constant area function. This function is better than Seco and the general equation. Most of the case the cylindrical surface machined in the same sequence so there is no runout problem and the extra material also not typical in CNC machining. So, the first safety pass is not necessary, which also helps to reduce the maximum load.



**Figure 20:** Infeed depth based on the optimal and suggested  $d$  values.

An iterative process to generate a database for degression factor by repeatedly applying the same method to obtain newly suggested degression factors that is customized for each metric thread pitch. The aim is to create a dataset that minimizes area differences based on Equation 10 when the thread is generated using EdgeCAM method, this dataset represented in Table 4.

**Table 4:** Optimal degression factors for all metric thread Pitches  $P$ .

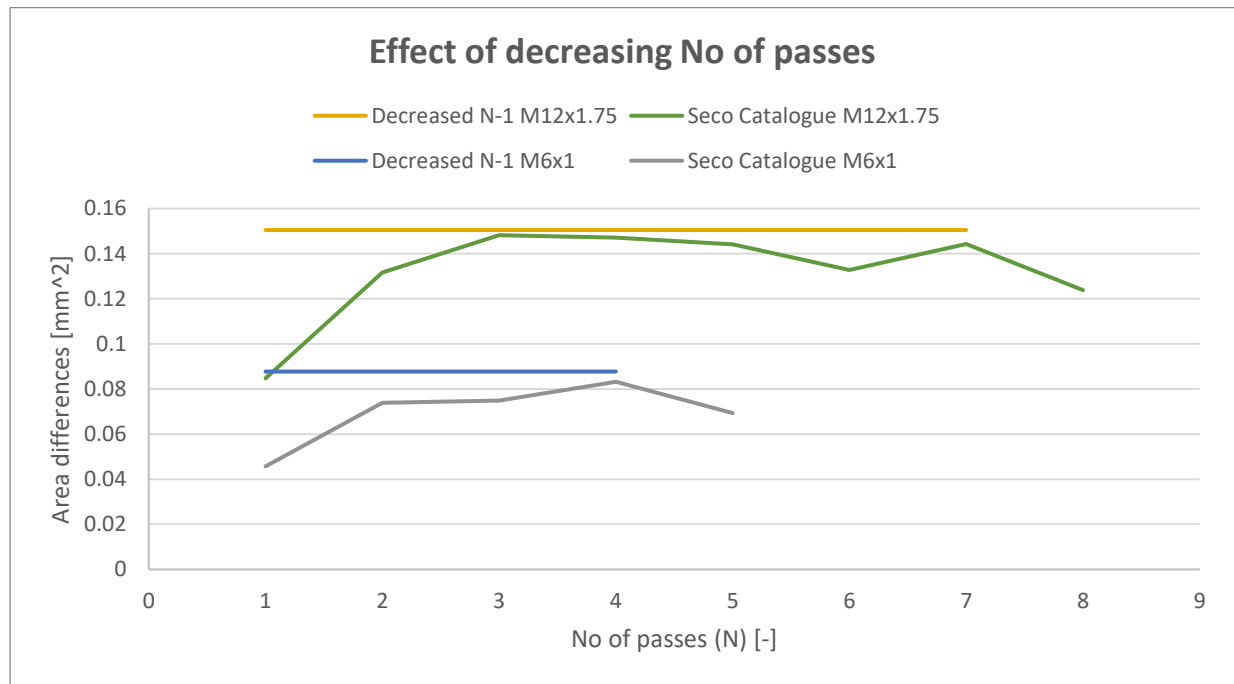
| $P$ [mm] | $d_{optimal}$ | $P$ [mm] | $d_{optimal}$ | $P$ [mm] | $d_{optimal}$ | $P$ [mm] | $d_{optimal}$ |
|----------|---------------|----------|---------------|----------|---------------|----------|---------------|
| 6        | 1.614         | 4        | 1.616         | 2        | 1.612         | 1        | 1.637         |
| 5.5      | 1.615         | 3.5      | 1.617         | 1.75     | 1.616         | 0.8      | 1.649         |
| 5        | 1.616         | 3        | 1.609         | 1.5      | 1.619         | 0.75     | 1.635         |
| 4.5      | 1.618         | 2.5      | 1.619         | 1.25     | 1.635         | 0.5      | 1.681         |

From this dataset it is observed that the optimal degression factor is always between 1.61 to 1.68 which are proved to be more efficient in generating less area difference in every machining cycle.

### 3.1.3. Effect of decreasing number of passes with constant area

Since the area is constant in all passes, this prevents it from overloading the insert, and this can be helpful to decrease the number of passes to increase the efficiency of the machining time. To test it the same parameters mentioned in Table 1 are used except the  $N$  which is the number of passes, it is decreased with one so  $N_{-1}$ , in that case a constant area will be obtained at every cycle  $A$ , that shall be compared with the maximum area suggested by Seco, if that area  $A$  doesn't exceed the 10% than the maximum area from Seco, it means that it can be safely used without overloading the insert.

In this case we have totally constant area, which means that we can study whether we can decrease the number of passes without overloading the insert or overload it with max of 5% than then maximum of what Seco suggests.



**Figure 21:** Difference between the decreased number of passes Compared to SECO catalogue.

In Figure 21 the effect of decreasing the number of passes to  $N_{-1} = (N - 1)$  is illustrated, in which it analyzed the area increase compared to the max area suggested by Seco. For M12x1.75 it has just 1.54% while for M6x1 it has a 5.5% difference, which is acceptable for both cases, that were calculated based on Equation 11.

$$Max\ area\ diffrnce\% = \frac{A_{N-1} - A_{max\ Seco}}{A_{max\ Seco}} \cdot 100\ \% , \quad (11)$$

### 3.1.4. Enhanced function for equal area

The insert used for the threading creates a full profile thread, which means that it creates the radius at the crest of the thread, and for this, the last two cycles will be affected with extra machining area. This extra area is based on the geometry of the insert as shown in Figure 11 (a) which corresponds to the filet radii in the insert. These filet radii are not recorded in the catalogues, only can measure with the microscope.

To avoid the extra measurement in case of every insert by the workers, I create the modification relative area according to the whole area. Total added area equivalent to 8%. This area is affecting the last two cycles with 70% in the last one, and 30% in the cycle before. For the best results of the thread quality, I will consider it to be 10 %, that will affect the last two cycles with 7.5 % in the last cycle and 2.5 % in the cycle before. That's why I will decrease the area in those two cycles and divide these values on the other cycles evenly as show in

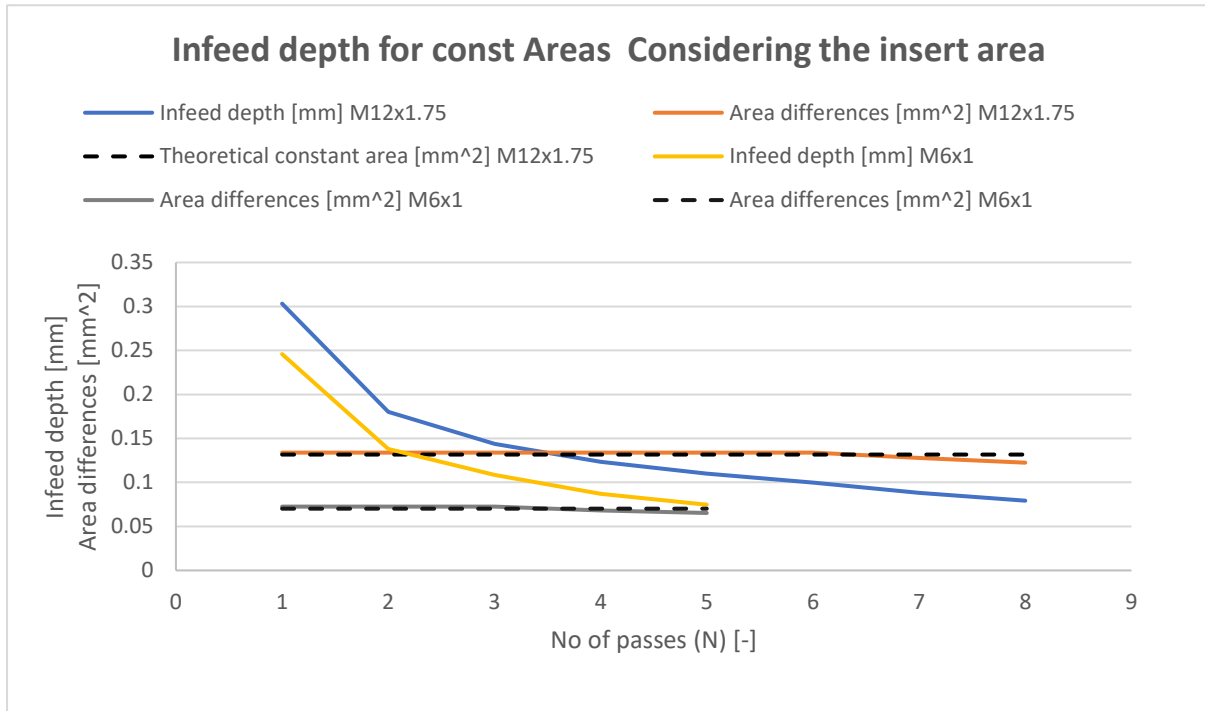
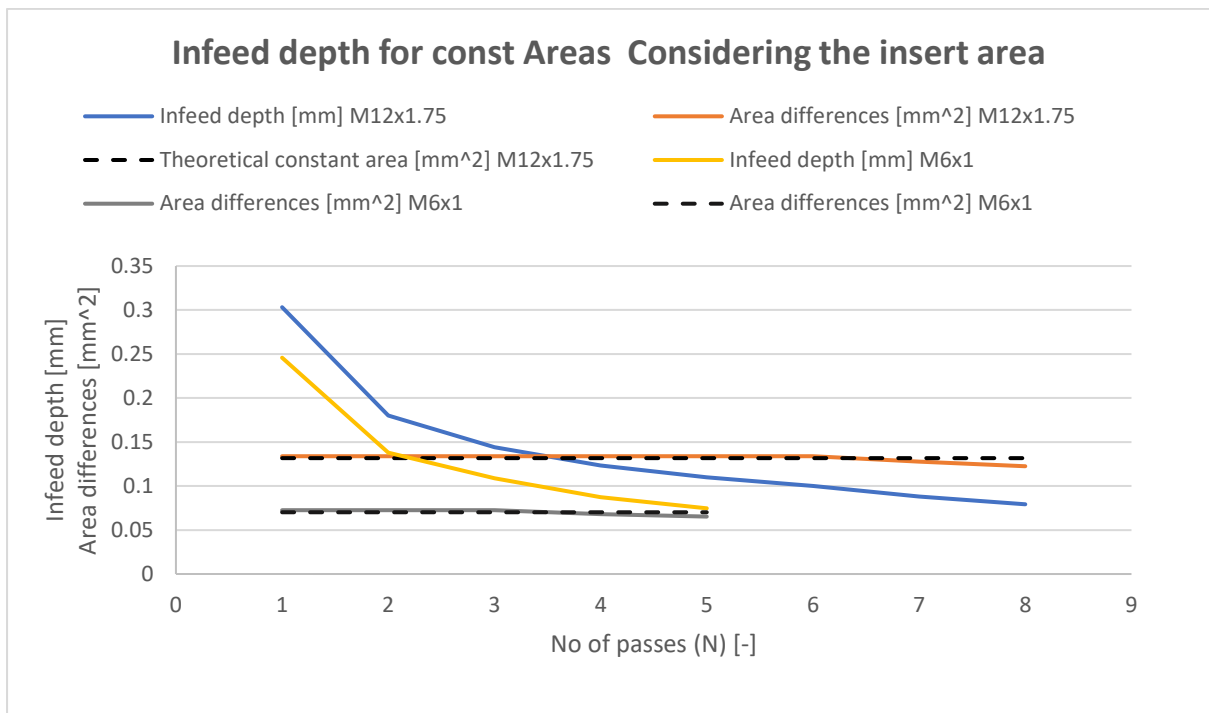


Figure 22. It can be observed that the area slightly increasing in the first 6 cycles then it decreases with 10 %, this will result in less force and better surface finish in the last cycle. This can help to avoid the extra path without any feed, which is a typical solution, if the thread quality is not appropriate.

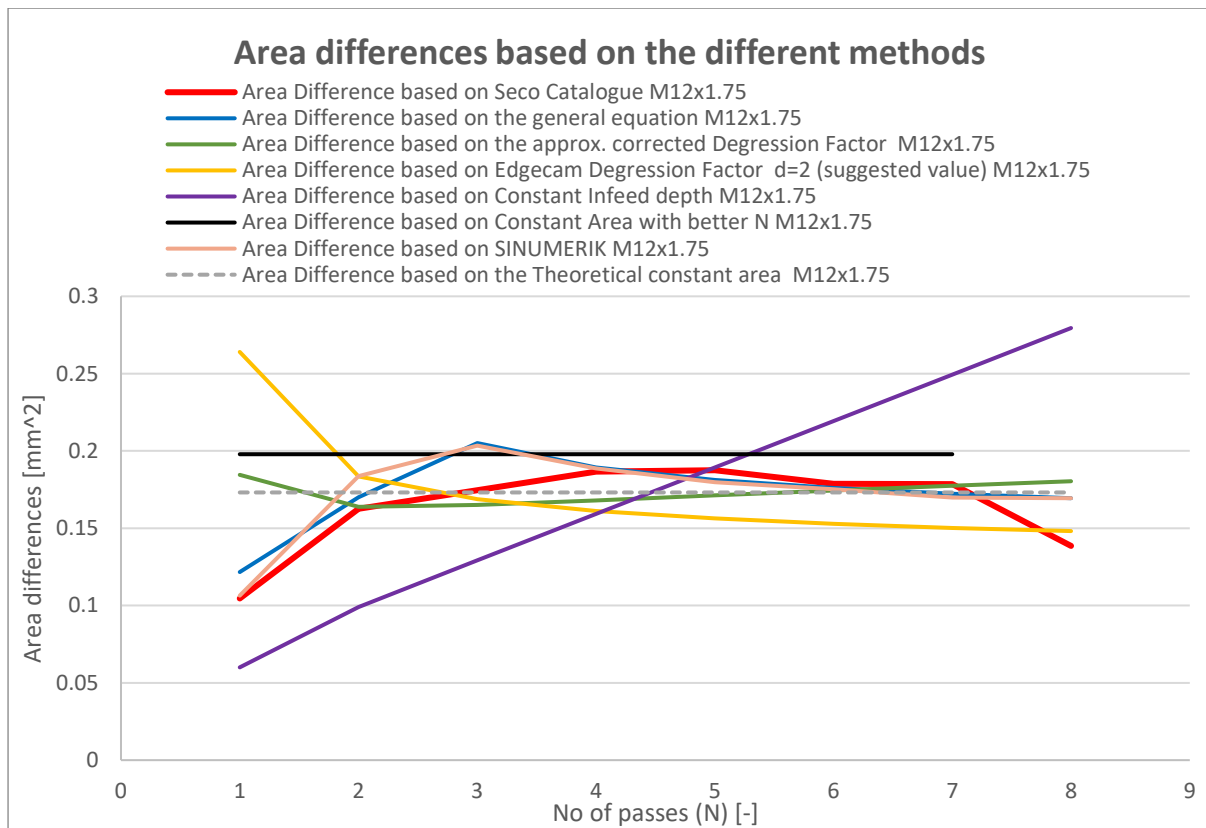


**Figure 22:** Suggested infeed depth.

### 3.1.5. Comparison of the different methods

After analyzing each method and understanding the different behavior of them, a comparison between these methods will explain it better, so Figure 23 contains the possible solution based on different methods.

1. Area differences based on Seco suggestions.
2. Area differences based on the general equation.
3. Area differences based on EdgeCAM suggested degression factor ( $d = 2$ ).
4. Area differences based on Corrected EdgeCAM degression factor ( $d = 1.62$ ).
5. Area differences based on theoretical constant area with reduced passes ( $N - 1$ ).
6. Area differences based on constant infeed depth.
7. Area differences based on the Sinumerik (Similar to the General Equation)
8. Area differences based on the Theoretical constant area.



**Figure 23:** Area differences based on the different methods for **M12x1.75**.

In summary, thread creation can be accomplished through either software-based methods like EdgeCAM and SINUMERIK 808D or by generating G-code using various approaches such as catalogue values, the general equation, constant infeed depth, or the newly derived function from Equation 9. Optimization of the degression factor in EdgeCAM resulted in improved area differences in each cycle, achieving a more efficient solution. Additionally, manually generating G-code with the derived function can reduce passes without overloading the insert, enhancing overall machining efficiency.

## 4. IT Framework

### 4.1. 'CNC Threading' application

In the development of the 'CNC Threading' application, two pivotal components were established to facilitate accurate and efficient computations, firstly defining the Database, then building the appropriate function using Python as a programming language.

The first step involved the creation of a dedicated database file. This repository serves as a reservoir for storing constant suggested values and pertinent standards. By centralizing this essential information, the database acts as a cornerstone, providing a reliable reference point for the application's intricate calculations.

Integral to the functionality of the application are Python functions meticulously crafted for this purpose. Among these, two functions stand out as paramount in which they are explained more in appendix (A, B):

- Function: *calc\_area(ap, radius, angle = theta)*
- Function: *calc\_infeed(area\_n, radius, angle = theta)*
- Function: *gcode\_g33(Zs, lt, total\_depth, runout = 0)*:

The screenshot displays the 'Thread Optimization' application window. It features a form for inputting thread parameters, a diagram of a thread profile, and a G-code output window.

**Input Parameters:**

- d: 12
- p: 1.75 (with a 'Standardize' button)
- R: 0.25
- ap: 1.128
- N: 8
- Theta: 60
- Basic Deviation: 9
- Tolerance Grade: 6

**Thread length [mm]:** 10

**Starting point (Z-axis):** 0

**ROP type:** LONG (selected from a dropdown menu)

**Run button:** A button labeled 'Run' is located below the input fields.

**Thread Diagram:** A technical drawing of a thread profile showing dimensions H, H/2, H/4, H/8, D, D<sub>2</sub>, D<sub>1</sub>, P, d<sub>3</sub>, d<sub>2</sub>, d, and angles of 60°.

**G-Code Output:**

```
T1 M16 D1;
G97 S200 M03;
G00 X13.0 Z2.0;      Cycle Starts !!
G00 X11.4005;
G33 Z-5.625 K1.75;
G33 Z-10.5 K12.5 K1.75;
G00 Z2.0;
G00 X11.0435;
G33 Z-5.625 K1.75;
G33 Z-10.5 K12.5 K1.75;
G00 Z2.0;
G00 X10.7539;
G33 Z-5.625 K1.75;
G33 Z-10.5 K12.5 K1.75;
G00 Z2.0;
G00 X10.5135;
G33 Z-5.625 K1.75;
G33 Z-10.5 K12.5 K1.75;
G00 Z2.0;
G00 X10.2958;
G33 Z-5.625 K1.75;
G33 Z-10.5 K12.5 K1.75;
G00 Z2.0;
G00 X10.0978;
G33 Z-5.625 K1.75;
```

Figure 24: CNC Threading application output for M12x1.75.

Figure 24 represents the GUI for the CNC Threading application in which Function *calc\_area* is responsible for computing the area based on specified parameters. It leverages mathematical transformations to accurately determine the area, incorporating considerations for the radius and angle.

While the *calc\_infeed* function plays a critical role in the application's functionality. It calculates the total depth, a crucial parameter in the threading process. By considering the specified area, radius, and angle, it delivers precise results, contributing to the application's overall accuracy.

Finally, the resulting G-code generated from this application was tested in *SINUMERIK808D on PC – V4.6* software that generates the tool path that is represented in Figure 25. In essence, the *gcode\_g33* function streamlines G-code generation for threading, providing adaptability and precision in CNC lathe machining processes.

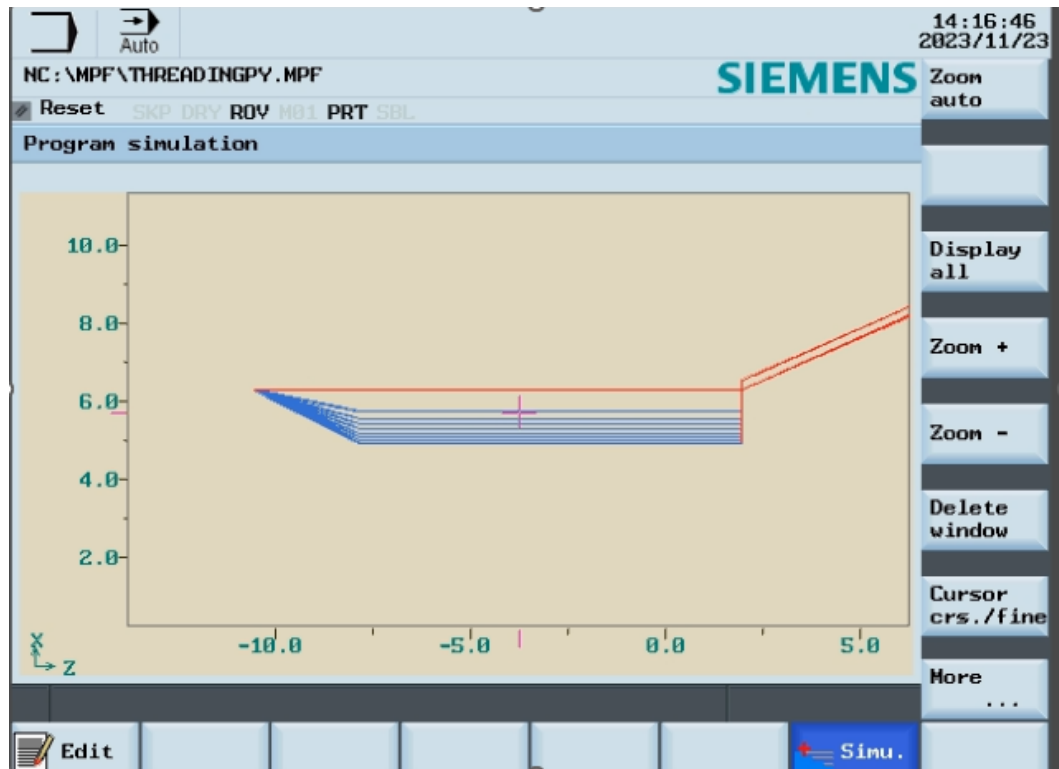
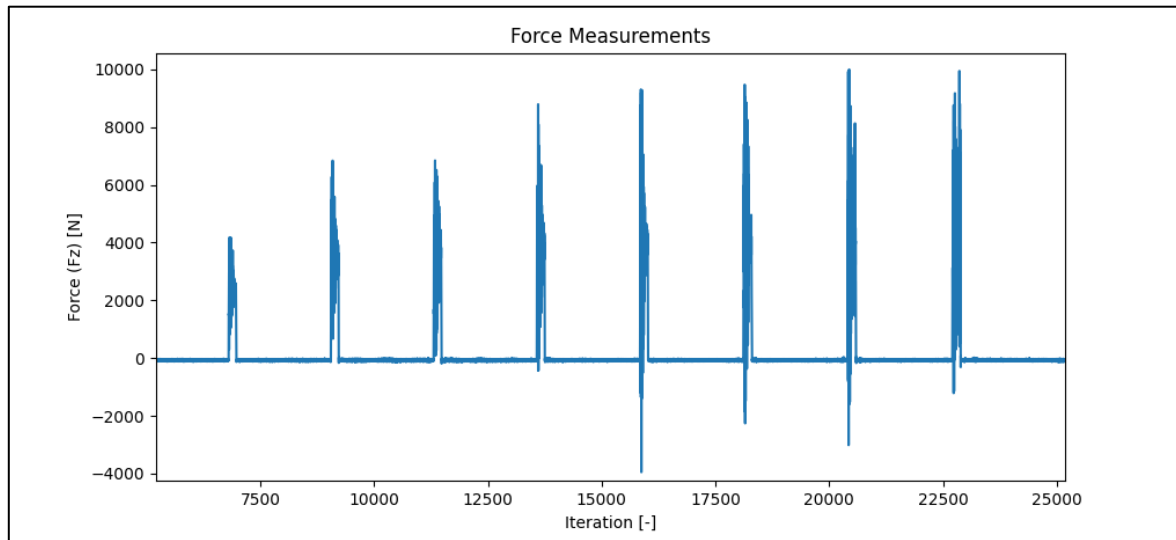


Figure 25: Tool path on SINUMERIK808D software.

After completing the application, a 16ER 1.75 ISO insert with a nose radius of 0.25 from TaeguTec was selected and the generated G-Code was used, and performed a threading operation on a steel workpiece that was pre-machined to a nominal diameter of 12 mm.



**Figure 26:** Force measurements for threading operation.

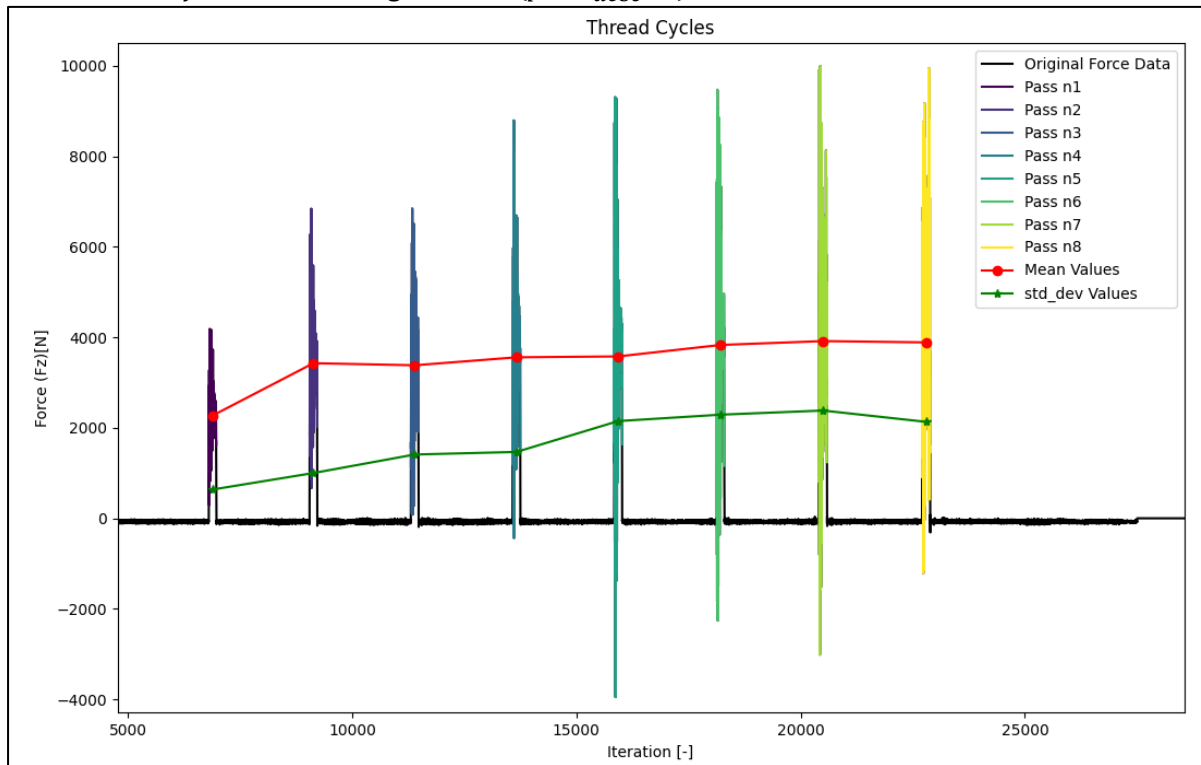
Subsequently, force measurements were process manually and illustrated in Figure 26. These measurements serve as a foundation for initiating the data analysis process, which aims to gain a deeper insight of the results. This analysis will be instrumental in making informed decisions regarding the enhancement process.

## 4.2. Force Analysis

Upon reviewing the force measurement graph seen in Figure 26, discerning the distinctive characteristics of the threading operation proves challenging. Some cycles exhibit significantly higher maximum forces, while others display heightened levels of vibration. As a preliminary measure, it becomes imperative to delineate the boundaries of each cycle. This delineation serves as the foundation for computing the mean and standard deviation values for every cycle, as exemplified in Figure 27. Furthermore, generating force distribution graphs, that are represented in Figure 28, provide valuable insights into the distribution patterns of each cycle.

Additionally, to facilitate this delineation process, two key functions were developed in a separate Python script that is explained in APPENDIX (B):

- Function: *detect\_machining\_phases*(*force\_data*, *window\_size* = 5, *threshold* = 200)
- Function: *find\_exceeding\_means*(*pass\_desc*, *X*)



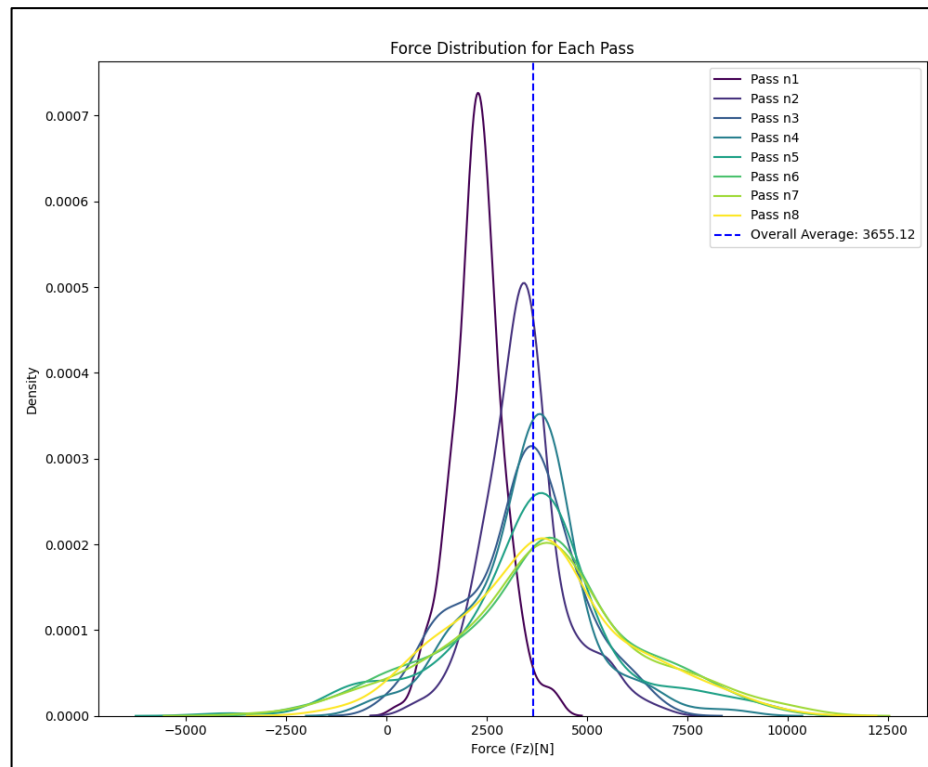
**Figure 27:** Machining cycles and information.

The first function employs a moving average filter to detect distinct machining phases within the force data. It considers parameters such as window size and threshold for precise phase identification. Additionally, it categorizes the force data into individual sections, enabling further analysis, those can be seen in the Thread Cycles graph in Figure 27. Also, it is used to plot the force distribution curves that is presented in Figure 28.

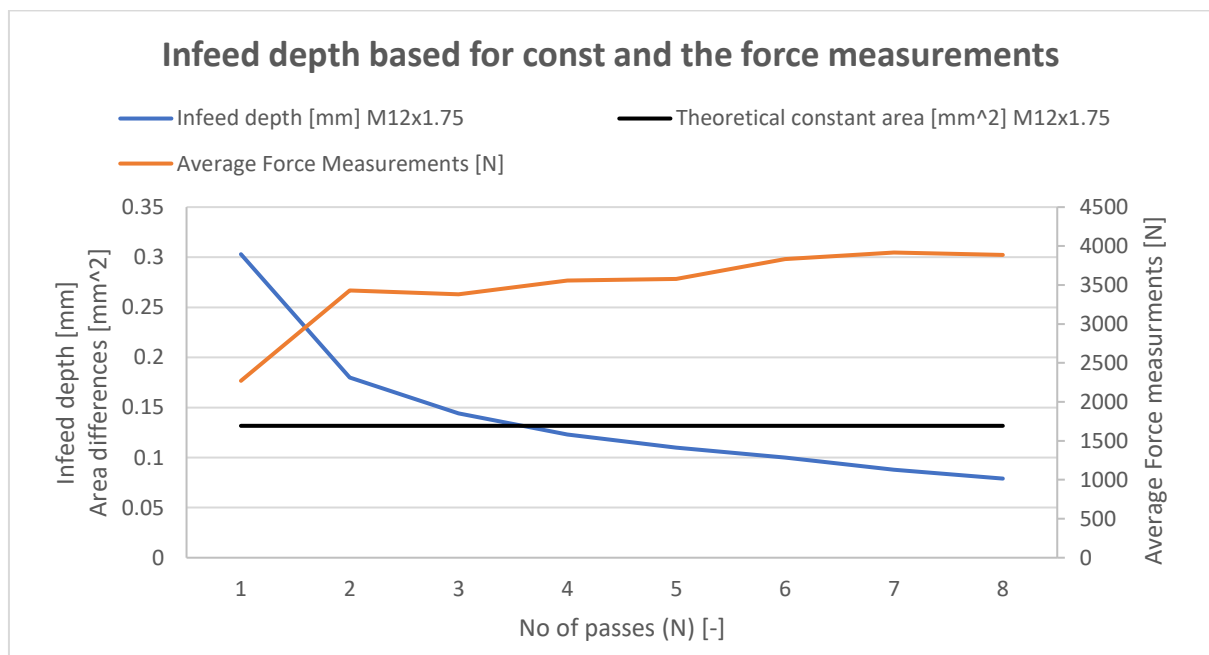
While the second function identifies mean values that surpass a defined percentage threshold *X*. It leverages descriptions provided in the *pass\_desc* dictionary. Furthermore, for comprehensive analysis, it calculates an overall average force, excluding the initial cycle as the

forces in it are independent of the equation rather it depends on the pre-machining of the workpiece.

These functions play a critical role in streamlining the analysis process, enabling a more granular understanding of force dynamics during the threading operation. This analytical groundwork lays the foundation for informed decision-making regarding operational parameters.



**Figure 28:** Force distribution graph.



**Figure 29:** Infeed depth, Theoretical constant area and Average Force measurements for **M12x1.75**.

Based on Figure 29, even the expected behavior should be constant force in each cycle but it is not the case, because the force tend to increase in the last two cycles. It can state that, the area changes in case for the least to pass is necessary as it was suggested in Section 3.1.4, this is due to two factors, the bigger vibration and the insert geometry.

a. Consistency in Average Forces

Figure 28 illustrates that the average forces across all cycles maintain remarkable consistency. Notably, the maximum average cycle force in does not deviate by more than 10% from the overall average. This uniformity in force distribution signifies a stable operational performance.

b. Correlation between Cycle Depth and Vibration

As the cycle number increases, indicating a deeper threading operation, a proportional increase in vibration levels becomes evident. This phenomenon is attributed to the augmented contact area between the insert and the workpiece.

c. Confirmation of Predicted Trends

Figure 27 validates the anticipated trend in average forces, aligning closely with the projections of the constant area in Figure 23. The nearly constant trajectory of average forces further reinforces the reliability of the 'CNC Threading' application's calculations.

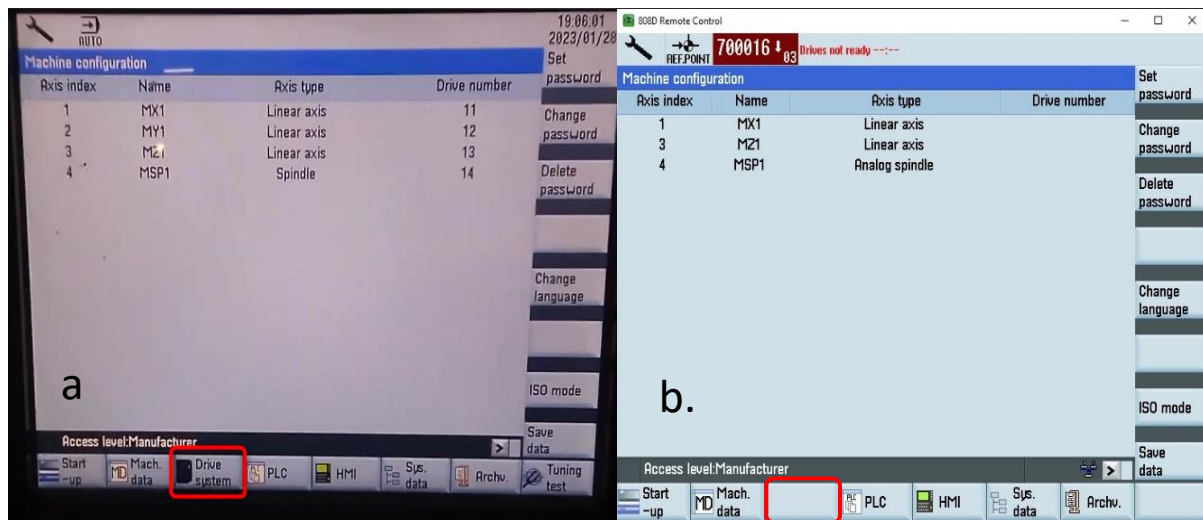
These findings not only provide critical insights into the behavior of forces during threading operations but also serve as a foundation for subsequent analyses and decision-making processes. The observed trends hold substantial implications for optimizing machining parameters and enhancing operational efficiency.

### **4.3. Automated Process**

Upon confirming the developed method, it was important to have real time observation for the thread operation on our CNC machines. This process requires a seamless and effective machine-to-PC communication and the ability to log essential data directly from the machine for subsequent analysis in the different python scripts. This section explores the challenges faced and the solutions implemented for automated data collection, it is performed on two machines, the EMCO Compact 5 (Sinumerik 808d) and the AKIRA CNC Lathe machine (Sinumerik 824d).

#### **4.3.1. EMCO Compact 5 (Sinumerik 808d)**

The EMCO Compact 5 (Sinumerik 808d) posed a challenge due to the absence of a direct option for accessing spindle torque readings, as illustrated by the comparison between the expected reference options and our machine in Figure 30.



**Figure 30:** Option to find the Spindle torque values. a) Reference b) Missing option from our machine

To address this limitation and enable data capture, an innovative solution was devised utilizing the *Access MyMachine* application, enabling screen mirroring on a connected computer. An Optical Character Recognition (*OCR*) model, implemented with the EasyOCR Python library, was employed to extract numeric values from a designated region of interest. This region could either be a fixed area or a dynamically defined box around the mouse pointer on the mirrored screen.

The process involves capturing screenshots within the specified box, extracting the recognized text, and retaining only numeric characters, dots, and negative signs. The outcomes from this OCR model can be saved in a preferred format for comprehensive data analysis. Moreover, the results can be displayed in real-time, providing a versatile solution for monitoring, as depicted in Figure 31.

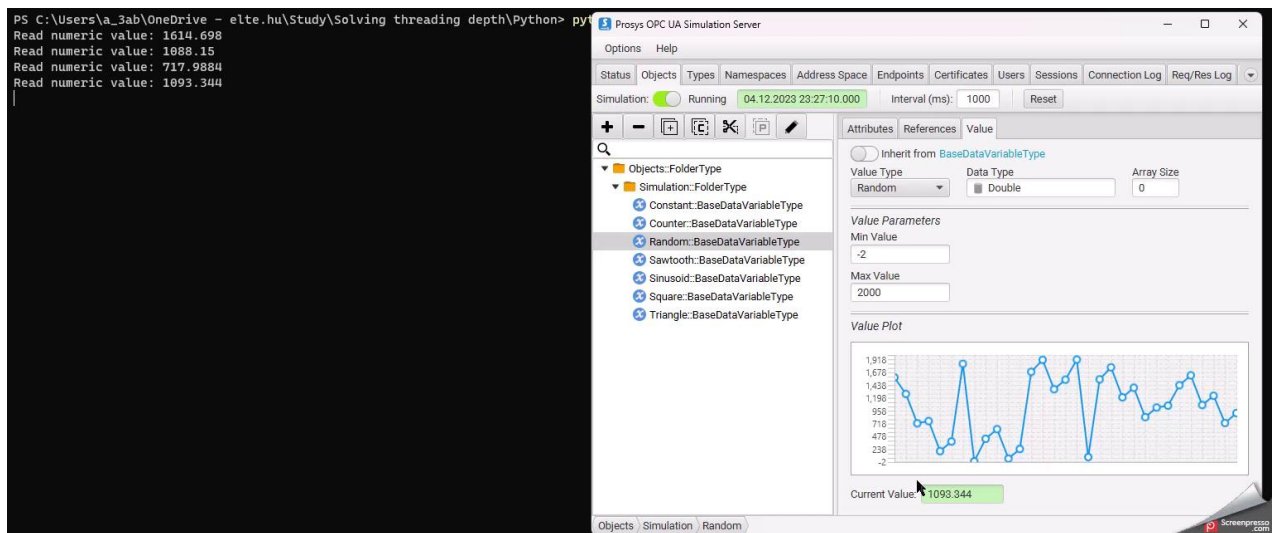


Figure 31: OCR model for EMCO machine.

### 4.3.2. AKIRA CNC Lathe Machine (Sinumerik 824d)

When encountering challenges in collecting necessary data from the EMCO machine, an exploration of the OPC UA protocol paved the way for successful communication with the AKIRA CNC Lathe machine. This machine, equipped with an OPC UA server as shown in Figure 32 that allowed for seamless connectivity through a properly configured OPC UA client. The client options were two-fold, presenting users with choices based on specific requirements.

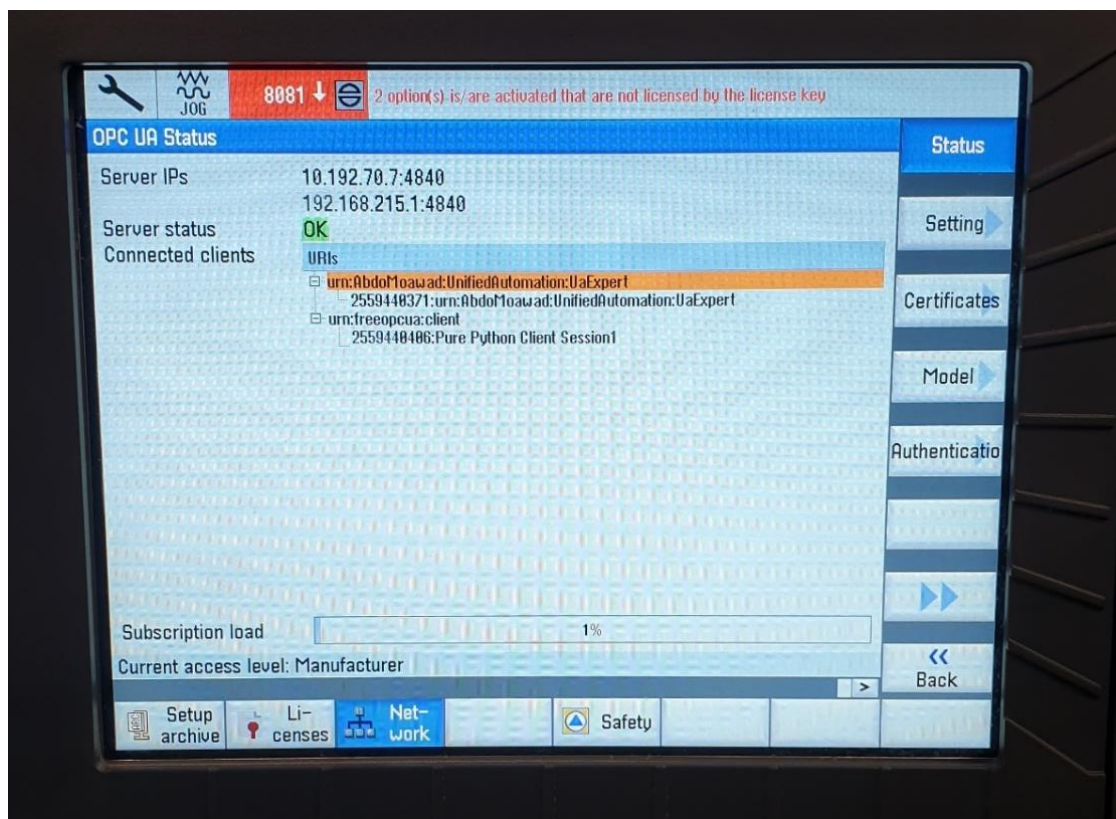
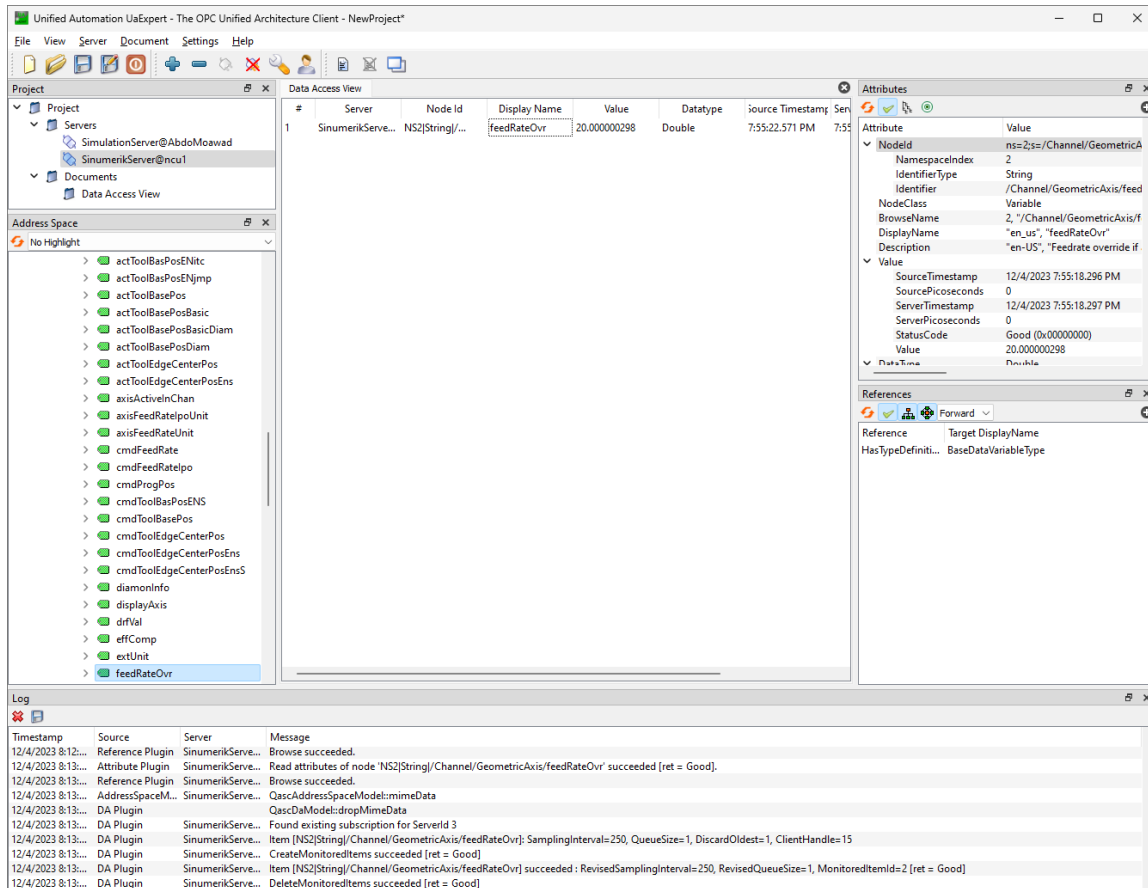


Figure 32: OPC UA Server on the Akira Lathe CNC machine.

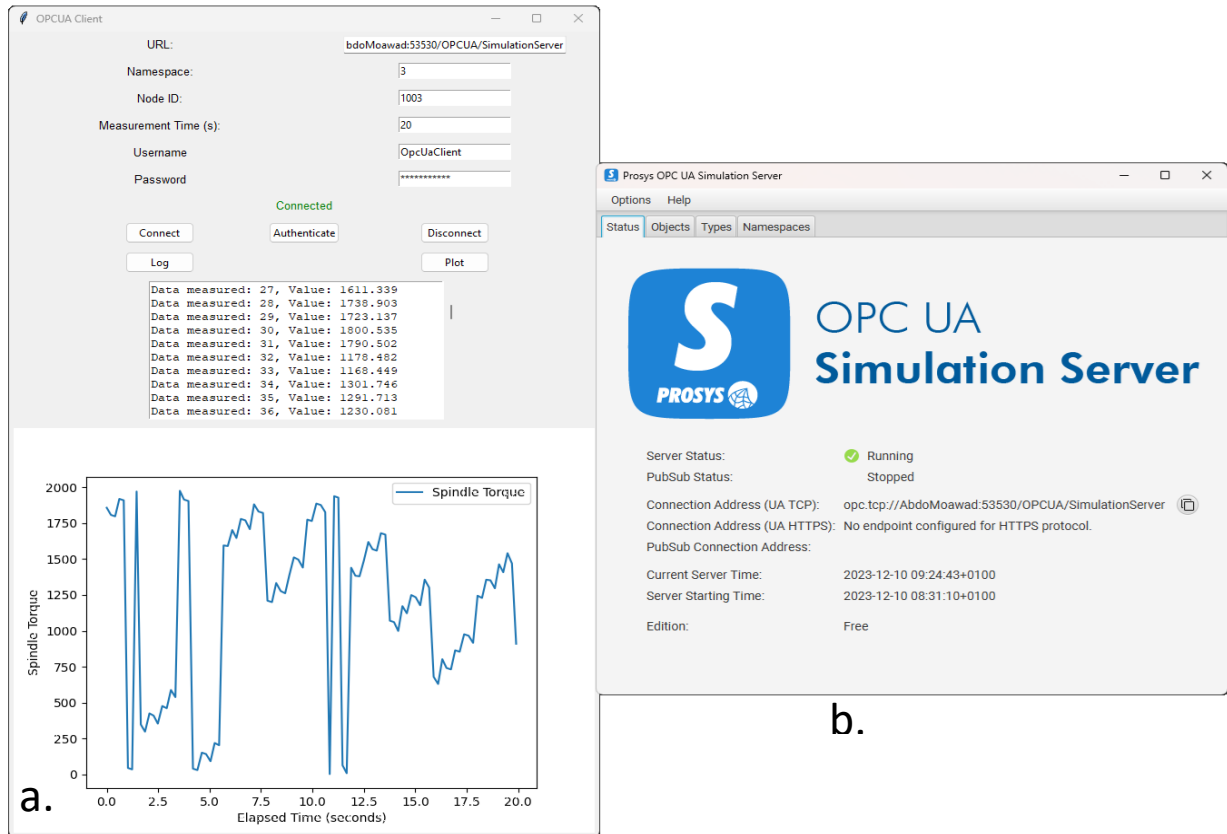
The first client solution involved utilizing the *UA Expert* software from Unified Automation, as illustrated in Figure 33. This software, which is free to use, provided a professional and user-friendly interface for OPC UA communication. Equipped with several built-in functions and a graphical user interface, UA Expert simplified the configuration process, enhancing user convenience.



**Figure 33:** UA Expert Client

The second client solution offered a high degree of customization through a custom Python application named *OPCUA Client* that is created as part of this thesis project, it is built based on the *opcua* library. This application, depicted in Figure 34, allowed users to define required nodes and customize the update frequency of measurements. The application facilitated logging and plotting of data, providing users with a comprehensive overview of the measured values.

Furthermore, the OPCUA Client Application incorporated a Graphical User Interface (GUI) for user-friendly interaction. The GUI included input fields for URL, namespace, node ID, measurement time, username and password. Users could effortlessly connect or disconnect from the server, authenticate with provided credentials, log data, and plot the recorded information, enhancing the overall usability of the application.



**Figure 34:** OPC-UA application a. OPCUA Client b. OPCUA Server simulator.

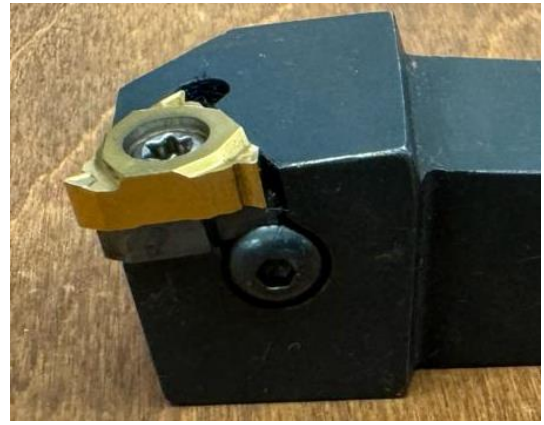
The client application worked by establishing a connection to the OPC UA server using the provided URL. Users could input the namespace and node ID to define the specific data to be accessed. The application offered functionalities to log and plot the measured values, contributing to a comprehensive understanding of the machine's performance over time.

The Akira machine is not equipped with a force sensor to read the force generated on the insert during the cutting process. To overcome this limitation, another parameter was targeted to have an indication for the force changing in every cycle, this parameter is the Spindle torque, as we expect that the cutting force will cause a higher spindle torque.

### 4.3.3. Experiment

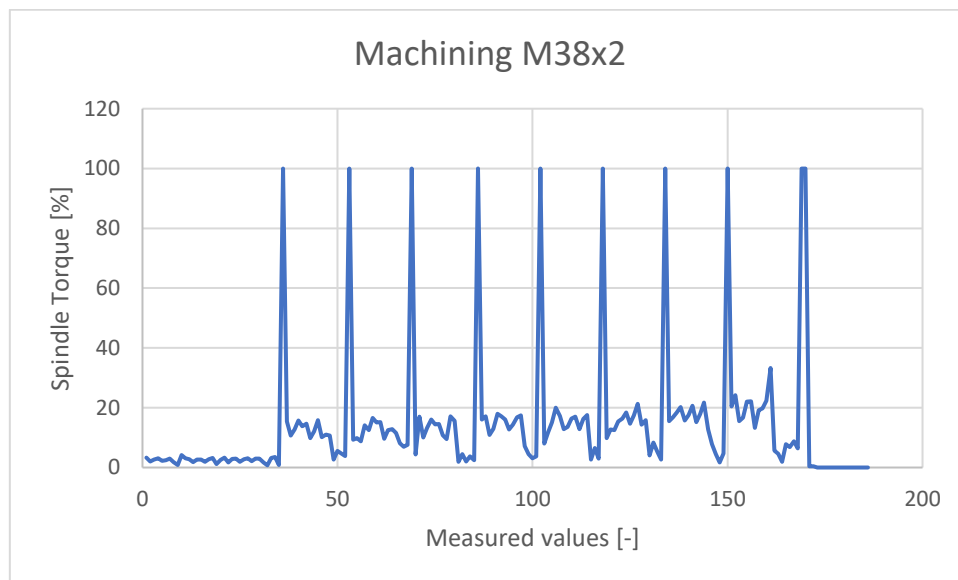
In our investigation, experiments aimed at enhancing thread operations on CNC lathe machines were conducted. The experimentation phase involved the utilization of three structural steel C15 workpieces, with the same thread pitch.

Two of the samples featured a diameter of 16 mm and a length of 50 mm, intended for the creation of an M16x2 thread. Conversely, the third sample possessed a diameter of 40 mm and a length of 130 mm, designated for an M38x2 thread. Employing a consistent insert, tool, and toolholder that is represented in Figure 35. All machining operations were conducted on the Akira CNC Lathe Machine using an auto generated G-code.



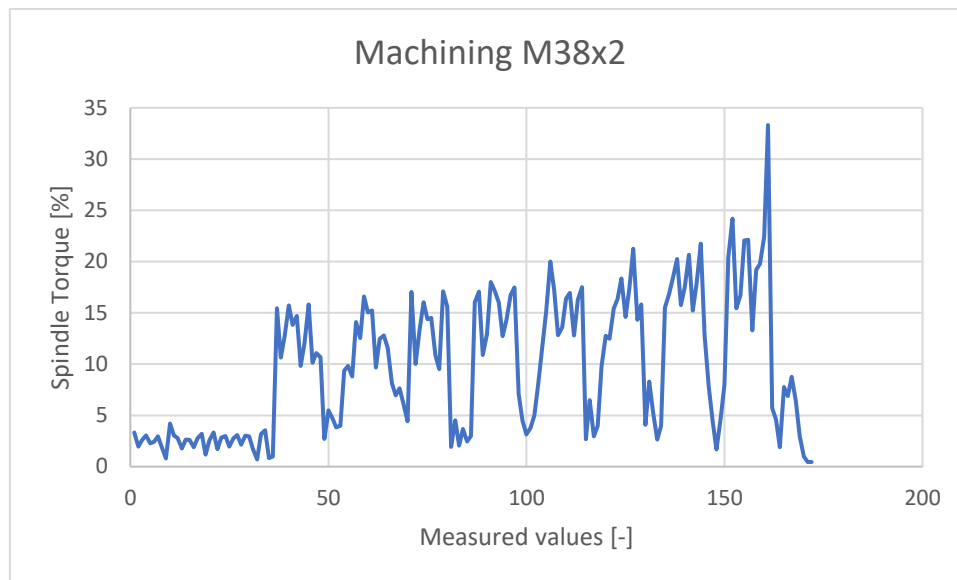
**Figure 35:** Threading Sample

To ensure precision and mitigate potential discrepancies, the workpiece with a 40 mm diameter underwent pre-machining to attain the requisite diameter of 38 mm. This preemptive measure aimed to preemptively address any deviations in outer diameter that could impede the threading process. Subsequently, threading operations were automated using the Thread Optimization application, yielding the torque measurements.



**Figure 36:** Torque measurement during threading operation.

Figure 36 represents the torque measurements obtained during the threading operation, representing the spindle's torque requirements to maintain the prescribed RPM. Notably, the graph reveals distinct patterns: prominent spikes occurring before each machining cycle, attributed to noise and disregarded for analysis and fluctuations delineating the machining and non-machining intervals, indicative of varying torque demands. These observations prompted a modification in the representation of torque measurements, as depicted in Figure 37 .



**Figure 37:** Modified torque torque measuremnt graph.

Figure 37 illustrates the refined torque measurements throughout the threading process, elucidating instances of torque percentage reduction denoting non-machining periods while the tool repositioning. The total behavior of the functions is similar as the expected values from Figure 27, where the force tends to increase in the last two cycles. However, it is noteworthy that fluctuations within certain cycles rendered the data unsuitable for precise analysis. Consequently, the incorporation of a load sensor emerges as imperative, ensuring accurate force measurements, particularly to account for area changes for each cycle and to enable a real-time monitoring of the system's performance.

## 5. Conclusions

In the field of thread cutting, numerous methods and models exist to achieve uniform force distribution. However, these methods often lack the desired level of accuracy. Catalogues, a common resource, may not encompass all the necessary data, necessitating efficient utilization of relative area measurements to overcome measurement-related challenges. The validation of force measurements emerges as a pivotal aspect, proving the efficacy of new methods compared to existing ones.

The initial pass in thread cutting is influenced by external factors, with a significant impact from the pre-machining of straight turning and finishing processes. Furthermore, it's crucial to note that as the number of passes increases, so does the potential for vibrations, which means that the last few cycles are affected with higher forces coming from this vibration, accordingly the machining area needs to be decreased in these cycles to overcome this extra force.

Regarding existing models, the thread creation landscape includes software built-in solutions such as EdgeCAM and SINUMERIK 808D, as well as G-code generation methods

involving catalogues, the general equation, constant infeed depth and the newly derived function. The optimization of the degression factor in EdgeCAM has notably enhanced area differences in each cycle, offering a more efficient solution. Additionally, manual G-code generation using the derived function proves advantageous by reducing passes without overloading the insert, thereby enhancing overall machining efficiency.

In the IT domain, our comprehensive data analysis endeavors, coupled with the functionality exhibited by the 'CNC Threading' application, have yielded outcomes in line with our initial projections. The application's adeptness in accurately determining cutting depth positions signifies a promising avenue for future optimization endeavors. This fusion of mechanical engineering principles with IT tools denotes a progressive stride towards enhancing precision and efficacy in threading operations conducted on CNC lathe machines.

In summary, our pursuit to automate threading processes on CNC lathe machines has yielded significant advancements, seamlessly amalgamating mechanical engineering with information technology. The latest iteration of our CNC threading application now features automated G-code generation, thereby refining threading operations and augmenting machining efficiency.

Addressing challenges presented by the EMCO Compact 5 (Sinumerik 808d), I have successfully implemented a data collection automation solution. Harnessing the capabilities of the Access MyMachine application, we deployed an Optical Character Recognition (OCR) model, powered by the EasyOCR Python library. This innovative approach facilitates the extraction of numerical values from the machine's interface, enabling real-time monitoring and versatile data acquisition.

Furthermore, the adoption of the OPC UA protocol marks a pivotal leap forward. Seamless communication with the AKIRA CNC Lathe machine, equipped with an OPC UA server, has been achieved. Through exploration of two client solutions UA Expert software and a custom Python application leveraging the opcua library we have ensured adaptability and customization in our approach. Additionally, the utilization of a simulator, 'prosys OPC UA simulation server', has provided a controlled environment for testing various client configurations.

In essence, our holistic automation strategy, spanning G-code generation, data collection optimization, and OPC UA integration, culminates in an advanced, efficient, and versatile solution tailored for threading operations on CNC lathe machines.

## 6. Short summary and ongoing work

In the intersection of mechanical engineering and information technology, the optimizing threading on CNC lathe machines poses a contemporary challenge. Mechanical engineering involves creating a Metric Thread adhering to ISO standards for consistency across machining cycles. On the IT side, a communication protocol aids data acquisition for informed determinations about insert integrity. The 'CNC Threading' app autonomously generates infeed depths for G-Code instructions based on threading parameters.

The project develops a new equation for infeed depth with constant threading area based on insert geometry. Various programming environments (G-code, Sinumerik, EdgeCAM) are compared for M6x1.0 and M12x1.75 to observe size effects. Results indicate necessary adjustments for the EdgeCAM degression factor and unique characteristics in Seco catalogue, general equation, and SINUMERIK software. Theoretical methods are practically applied using the 'CNC Threading' application and analyzed with a Python script.

In terms of Automation work, the focus is on Automated G-Code Generation, aiming to streamline programming and reduce manual intervention. Additionally, integrating with the AKIRA CNC Lathe Machine offers an opportunity for real-time spindle torque measurements, providing insights into cutting process dynamics.

The ongoing initiatives aim to refine and expand application capabilities to improve the performance related to the data collected from the Akira Lathe Machine through the OPC UA port and prepare it for further analysis to have a real-time insight about the performance, all of this ushering in a new phase of optimized threading operations on CNC lathe machines, promising enhanced efficiency in threading processes.

**Keywords:** *Cutting Tool lifetime, Digital communication, Python, Smart Manufacturing, Threading force*

## 7. References

- [1] K. Schwab, "The fourth industrial revolution, Crown Business," *New York*, vol. 192, 2017.
- [2] J. H. Kim, "A review of cyber-physical system research relevant to the emerging IT trends: industry 4.0, IoT, big data, and cloud computing," *Journal of industrial integration and management*, vol. 2, no. 03, p. 1750011, 2017.

- [3] H. Kagermann, W.-D. Lukas, and W. Wahlster, "Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution," *VDI nachrichten*, vol. 13, no. 1, pp. 2–3, 2011.
- [4] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & information systems engineering*, vol. 6, pp. 239–242, 2014.
- [5] A. Jamwal, R. Agrawal, M. Sharma, and A. Giallanza, "Industry 4.0 technologies for manufacturing sustainability: a systematic review and future research directions," *Applied Sciences*, vol. 11, no. 12, p. 5725, 2021.
- [6] E. Geisberger and M. Broy, *agendaCPS: integrierte forschungsagenda cyber-physical systems*, vol. 1. Springer-Verlag, 2012.
- [7] OPC Foundation, "Unified Architecture - OPC Foundation ." Accessed: Oct. 29, 2023. [Online]. Available: <https://opcfoundation.org/about/opc-technologies/opc-ua/>
- [8] mcu.by, "Industrial Network Protocols. OPC UA. Part 1." Accessed: Oct. 29, 2023. [Online]. Available: <https://www.mcu.by/opc-ua-part-1/>
- [9] J. Xiong, X. Xiong, X. Li, and B. Yu, "The research of CNC communication based on industrial Ethernet," in *Electrical Engineering and Control: Selected Papers from the 2011 International Conference on Electric and Electronics (EEIC 2011) in Nanchang, China on June 20-22, 2011, Volume 2*, Springer, 2011, pp. 689–694.
- [10] "Siemens S7", Accessed: Oct. 29, 2023. [Online]. Available: [https://cache.industry.siemens.com/dl/files/115/82212115/att\\_108330/v2/82212115\\_s7\\_communication\\_s7-1500\\_en.pdf](https://cache.industry.siemens.com/dl/files/115/82212115/att_108330/v2/82212115_s7_communication_s7-1500_en.pdf)
- [11] IPCOMM GmbH, "S7 Protocol (RFC 1006)." Accessed: Nov. 08, 2023. [Online]. Available: <https://www.ipcomm.de/protocol/S7ISOTCP/en/sheet.html#REF1>
- [12] S. Coromant, "Metalcutting technical guide," *Singapore: Sandvik Coromant*, 2005.
- [13] Y. Altintas, M. Khoshdarregi, and M. R. Khoshdarregi, "Generalized modeling of chip geometry and cutting forces in multi-point thread turning," 2015, doi: 10.13140/RG.2.1.4186.5445.
- [14] T. Suzuki, Y. Ariura, and Y. Umezaki, "Basic Study on Cutting Forces in Gear Cutting : Theory of Cutting with Two Continuous Symmetrical Cutting Edges," *JSME international journal. Ser. C, Dynamics, control, robotics, design and manufacturing*, vol. 36, no. 4, pp. 543–548, 1993, doi: 10.1299/jsmec1993.36.543.
- [15] Y. UMEZAKI, Y. FURUMIYA, and T. SUZUKI, "The Numerical Analysis of Hobbing Torques : 1st Report, On the hobbing torques in various hobbing methods," *Bulletin of JSME*, vol. 26, no. 215, pp. 897–904, 1983, doi: 10.1299/jsme1958.26.897.

- [16] W. König and K. Bouzakis, "Chip formation in gear-shaping," *Ann. CIRP*, vol. 26, no. 1, pp. 17–20, 1977.
- [17] G. Fromentin and G. Poulachon, "Geometrical analysis of thread milling—part 1: evaluation of tool angles," *The International Journal of Advanced Manufacturing Technology*, vol. 49, pp. 73–80, 2010.
- [18] G. Fromentin and G. Poulachon, "Geometrical analysis of thread milling—part 2: calculation of uncut chip thickness," *The International Journal of Advanced Manufacturing Technology*, vol. 49, pp. 81–87, 2010.
- [19] A. C. Araujo, J. L. Silveira, M. B. G. Jun, S. G. Kapoor, and R. DeVor, "A model for thread milling cutting forces," *Int J Mach Tools Manuf*, vol. 46, no. 15, pp. 2057–2065, 2006.
- [20] M. B. G. Jun and A. C. Araujo, "Modeling of the thread milling operation in a combined thread/drilling operation: Thrilling," 2010.
- [21] Y. Altintas and A. A. Ber, "Manufacturing automation: metal cutting mechanics, machine tool vibrations, and CNC design," *Appl. Mech. Rev.*, vol. 54, no. 5, pp. B84–B84, 2001.
- [22] H. K. Akyildiz and H. Livatyali, "Prediction and experimental analysis of cutting forces during machining of precision external threads," *Machining science and technology*, vol. 15, no. 4, pp. 485–506, 2011.
- [23] H. K. Akyildiz, "Evaluating of cutting forces in thread machining," *The International Journal of Advanced Manufacturing Technology*, vol. 68, pp. 1601–1612, 2013.
- [24] F. Sineider Martins, R. Reina-Muñoz, and V. Martins Lira, "System of cutting force data acquisition in mechanical lathes," *Dyna (Medellin)*, vol. 85, no. 207, pp. 16–21, 2018.
- [25] Sandvik Coromant, "Sandvik Threading formulas and definitions." Accessed: Nov. 08, 2023. [Online]. Available: <https://www.sandvik.coromant.com/en-gb/knowledge/machining-formulas-definitions/threading-formulas-definitions>
- [26] V. M. Lira, "Cut parameters optimization in the threading machining on CNC Lathes," MSc. Dissertation, Campinas, Brazil, 2000.
- [27] G. Fredericks, "A threading algorithm suitable for computer numerically controlled turning centers," *International Journal of Machine Tool Design and Research*, vol. 18, no. 1, pp. 29–31, 1978.
- [28] Seco, "STANDARD PRODUCTS." [Online]. Available: [www.secotools.com](http://www.secotools.com).
- [29] B. ISO, "965-1: 2013–ISO general purpose metric screw threads-Tolerances Part 1: Principles and basic data," *BSI, London, UK*, 2013.

# APPENDIX

## A. Insight for ‘CNC Threading’ software.

```
import math

def calc_area(ap, radius, angle):
    """
    Calculates the area of a specific cutting depth.

    Args:
        ap (float): Total cutting depth.
        radius (float): Nose Radius.
        angle (float): Thread Angle in degrees = 60 degrees.

    Returns:
        float: Calculated area.
    """
    angle_rad = math.radians(angle)
    area = (((ap + radius) ** 2 - radius ** 2 *
             (3 - math.pi / 3 * math.tan(angle_rad))) / math.tan(angle_rad))
    return area

def calc_infeed(area_n, radius, angle):
    """
    Calculates the total depth required for machining area.

    Args:
        area_n (float): Total machining Area.
        radius (float): Nose Radius.
        angle (float): Thread Angle in degrees = 60 degrees.

    Returns:
        float: Calculated total depth based on the required machining area.
    """
    angle_rad = math.radians(angle)
    total_depth = math.sqrt(area_n * math.tan(angle_rad) + radius ** 2
                             * (3 - math.pi / 3 * math.tan(angle_rad))) - radius
    return total_depth
```

**Figure 38:** Code snippet for CNC Threading Software important functions.

Figure 38 is a code snippet outlining the main functions of the **CNC Threading** application, leveraging Equations 8 and 9 for thread optimization.

## B. Automated G-code generation

```
import math

def gcode_g33(Zs, lt, total_depth, runout=0):
    """
    Generate G-code for threading on Siemens Sinumerik controller using G33.

    Parameters:
        Zs (float): Starting point for the thread.
        lt (float): Thread length including the runout.
        total_depth (list[float]): Total depth of insert (multiple depths).
        runout (int): Runout type (0: no runout, 1: long runout [2.5xP], 2:
short runout [1.25xP]).

    Returns:
        str: Generated G-code string.
    """
    global d, p

    if runout == 'NO':
        l = lt
    elif runout == 'LONG':
        l = lt - 2.5 * p
    elif runout == 'SHORT':
        l = lt - 1.25 * p

    gcode = "T1 M16 D1;          Tool selection\n"
    gcode += "G97 S200 M03;      Spindle Speed\n"
    gcode += f"G0 X{d + 1} Z{Zs + 2};      Cycle Starts !!\n"

    for depth in total_depth:
        gcode += f"G00 X{(d - depth * 2):.2f};\n"
        gcode += f"G33 Z-{l} K{p};\n"
        gcode += f"G33 Z-{lt + 0.5} X{d + 0.5} K{p};\n"
        gcode += f"G00 Z{Zs + 2};\n"

    gcode += "G00 X100 Z100;\n"
    gcode += "M30"

    return gcode
```

Figure 39: Automated G-code generation "G33 cycle."

### C. Data Analysis Python scripts

```
import numpy as np
import pandas as pd

def detect_machining_phases(force_data, window_size=5, threshold=200):
    """
        Detects machining phases in force data using a moving average filter.

    Args:
        force_data (pandas DataFrame): DataFrame containing force measurements.
        window_size (int): Size of the moving average window.
        threshold (float): Threshold for detecting machining phases.

    Returns:
        machining_phases (list[tuple]): A list of tuples representing the start
and end indices of machining phases.
        sections (dict): A dictionary containing the values during the machining
processes.
        pass_desc (dict): A dictionary containing descriptions of each cycle.
    """
    force_values = force_data['Fz'].to_numpy()
    moving_avg = np.convolve(force_values, np.ones(window_size) / window_size,
mode='valid')
    start_indices = []
    end_indices = []

    for i in range(len(moving_avg) - window_size + 1):
        if abs(moving_avg[i]) > threshold:
            if i == 0 or abs(moving_avg[i - 1]) <= threshold:
                start_indices.append(i + window_size)
            if i == len(moving_avg) - 1 or abs(moving_avg[i + 1]) <= threshold:
                end_indices.append(i - 1)

    machining_phases = list(zip(start_indices, end_indices))

    sections = {} # Create a dictionary to hold sections
    pass_desc = {}

    for i, (start, end) in enumerate(machining_phases):
        section_name = f'n{i + 1}' # Generate section name
        section_data = force_data.iloc[start:end + 1]
        sections[section_name] = section_data

        mean_value = section_data['Fz'].mean()
        min_value = section_data['Fz'].min()
        max_value = section_data['Fz'].max()
        index_co = (section_data.index.start + section_data.index.stop) // 2
        std_dev = section_data['Fz'].std() # Calculate standard deviation
```

```
pass_desc[section_name] = {
    'mean': mean_value,
    'min': min_value,
    'max': max_value,
    'index': index_co,
    'std_dev': std_dev
}

return machining_phases, sections, pass_desc
```

**Figure 40:** Code explanation for *detect\_machining\_phases* function.

Figure 40 elucidates the purpose, required arguments, and output of the *detect\_machining\_phases* function. This function employs a moving average filter, calculating averages for a specific number of elements. If no machining is detected, the calculated value remains below the determined threshold. Once the value surpasses this limit, it signals the start of a machining cycle. When it drops below the threshold again, it signifies the end of that machining cycle. The function returns detailed information for each pass, including maximum, minimum, average, and standard deviation values, crucial for subsequent analyses.

```

def find_exceeding_means(pass_desc, X):
    """
    Finds mean values that exceed a certain percentage `X` from the
    `pass_desc` dictionary.

    Args:
        pass_desc (dict): Dictionary containing pass descriptions.
        X (float): Percentage threshold for exceeding means.

    Returns:
        exceeding_means (list[tuple]): List of tuples (section_name,
        mean_value) where mean_value exceeds `X` percent.
        overall_average (float): The value of the average force from all
        passes regardless of the first pass.
    """
    exceeding_means = []

    # Calculate the overall average of mean values (ignoring the first cycle)
    all_mean_values = [details['mean'] for details in pass_desc.values()][1:]
    overall_average = sum(all_mean_values) / len(all_mean_values)

    for section_name, details in pass_desc.items():
        mean_value = details['mean']
        if mean_value > (1 + X / 100) * overall_average:
            exceeding_means.append((section_name, mean_value))

    return exceeding_means, overall_average

```

Figure 41: Code explanation for *find\_exceeding\_means* function.

Figure 41 represents the function *find\_exceeding\_means*. This function relies on the average mean value of all cycles obtained from the *pass\_desc* parameter in the previous function. If any mean value exceeds a predetermined threshold percentage denoted as *X*, it is flagged as a high-force cycle. Further analysis is conducted to avoid such cycles during subsequent machining with the same insert.

## D. OCR Python Script

```
def process_image(image):
    """
    For demonstration, this function converts the image to grayscale.
    Args:
        image (numpy.ndarray): Input image (BGR format).
    Returns:
        numpy.ndarray: Processed grayscale image.
    """
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    return gray_image

def read_screen_values():
    """
    Reads screen values around the mouse position and processes them.
    Returns: A list[float]: List of values (RPM) from the defined screen regions.
    """
    try:
        RPM = []
        while True:
            # Get the current mouse coordinates
            mouse_x, mouse_y = pyautogui.position()

            # Define the screen region around the mouse position
            roi_width, roi_height = 200, 200
            x = max(0, mouse_x - roi_width // 2)
            y = max(0, mouse_y - roi_height // 2)

            # Capture the screenshot of the defined region
            screenshot = pyautogui.screenshot(region=(x, y, roi_width,
            roi_height))

            # Convert the screenshot to a NumPy array (OpenCV format)
            frame = np.array(screenshot)

            # Process the image using the image processing logic
            processed_image = process_image(frame)

            average_pixel_value = np.mean(processed_image)
            RPM.append(average_pixel_value)
            time.sleep(0.01)

        except KeyboardInterrupt:
            print("Screen reading stopped.")

    return RPM
```

Figure 42: Python script using the OCR.