Genuense Athenaeum

Università degli Studi di Genova

Scuola Politecnica & Facoltà di Ingegneria

Laurea Magistrale in Ingegneria Strategica

International MSc in Engineering Technology for Strategy & Security

of Genoa University

Mechanical drawing feature extraction for automating test report generation within Industry 4.0 framework at Ansaldo Energia

Advisor(s): **Prof. Luca Mantelli (UNIGE)**

Prof. Alessandro Sorce (UNIGE)

Eng. Fulvio Giannoni (Ansaldo Energia)

Candidate: **Amine Messai**

Academic Year 2025/2026

Abstract

In the era of Industry 4.0, manufacturing processes increasingly demand automated, data-driven quality assurance to bridge the gap between design intent and production reality. Mechanical engineering drawings, as the definitive blueprints for precision components, encode critical dimensions, tolerances, and Geometric Dimensioning and Tolerancing (GD&T) symbols essential for inspection and compliance. However, manual extraction of this information remains a labor-intensive bottleneck, prone to errors and delays, particularly in high-stakes sectors like power generation. This thesis develops and validates a novel hybrid intelligence software tool for automating quality reporting by extracting and correlating dimensional data from paired CAD files and PDF drawings. The tool structurally prevents transcription errors through a mandatory human-in-command selection workflow, where only user-confirmed elements enter the final output.

The developed tool is a standalone desktop application built in Python, combining a Flask/pywebview GUI with a multi-stage processing backend. The pipeline comprises: deterministic IGES parsing to tokenize dimensions and tolerances; vector-text extraction from PDFs via PyMuPDF capturing spatial and semantic context; a heuristic matching algorithm scoring candidate text groupings by proximity, orientation, and reading order; targeted computer vision via OpenCV template matching for graphical symbols in programmatically defined search regions; and LLM-based classification of GD&T feature control frames using engineered prompts with few-shot examples and disambiguation rules. Feature control frames are isolated as image snippets for user review. The GUI enables intuitive cropping, annotation, and Excel report generation that mirrors pre-existing inspection templates.

Validated through a case study at Ansaldo Energia using 10 diverse technical drawings, the system demonstrated 98% accuracy in automated suggestions, high final report accuracy by architectural design, and a drastic reduction in report generation time. This work contributes a transparent, cost-effective alternative to opaque commercial APIs, redefining human-AI collaboration as "human-in-command" for reliable, scalable quality assurance in Model-Based Enterprises.

Dedication

My deepest gratitude is directed towards God, whose providence and grace have been the foundation of this journey. For the strength to overcome challenges, the wisdom to navigate complexities, and the remarkable individuals He placed along my path, I am profoundly thankful.

This work is also dedicated to my beloved family and friends. Your unwavering love, encouragement, and sacrifices have been my constant inspiration and a source of immeasurable support. This achievement was made possible by your belief in me, and for that, I am forever grateful.

Acknowledgement

I wish to express my deepest appreciation to my advisors, Professor Luca Mantelli and Professor Alessandro Sorce. Their continuous mentorship, insightful guidance, and rigorous feedback were instrumental in shaping the direction and quality of this research.

My sincere thanks are also extended to my Company tutor, Eng. Fulvio Giannoni. His industry perspective and practical knowledge were crucial in bridging the gap between theory and application. I am also profoundly grateful for the support of the entire Company team.

I would like to acknowledge my colleague, Brahim Bouzida, for the camaraderie and collaborative spirit that made this journey more enriching.

Finally, I owe a profound debt of gratitude to my family and friends. Their unwavering patience, encouragement, and belief in me were my foundation during the most demanding periods of this work.

Table of Figures

Figure I-1 Example of an engineering drawing 23
Figure II-1 https://reformcreative.co.uk/vector-vs-raster-files/ [8] 16
Figure II-1 Example of potential misidentifications 20
Figure II-2 Examples of Visually Similar characters and GD&T Symbols 20
Figure II-3 Template Matching Illustration [14] 30
Figure III-1 Example of attempted OCR using Tesseract .**Error! Bookmark not defined.**
Figure III-2 Detailed System Flowchart Illustrating the Multi-Stage Pipeline 36
Figure III-3 Template Matching with NMS example 38
Figure III-4 Illustration of Dynamic Search Area Calculation 39
Figure IV-1 Initial Interface Screenshot 43
Figure IV-2 Interface Crop View with LLM suggested annotation 44
Figure IV-3 Interface Crop View with Dimension Annotation 45
Figure IV-4 Adding Tolerance Interface 46
Figure IV-5 Debugging Step - Sample Token Output for Test IGES File 48
Figure IV-6 Extracted Spans for Sample Page with Angle Calculations 49
Figure IV-7 Debugging Step - Rendered Region and Detection Visualization 50
Figure IV-8 Example of a Detection Visualization 51
Figure IV-9 Example of a Returned Value from the Gemini LLM API

Content

Abstract 3
Dedication 4
Acknowledgement 5
Table of Figures 6
Abbreviations 10
General Introduction 11
Chapter I. Introduction 12
A. Introduction to Quality Assurance in Modern Manufacturing 12
1. The Industry 4.0 Imperative for High-Precision Manufacturing 12
2. The Engineering Drawing as the Primary Reference Document **Error! Bookmark not defined.**
3. The Manual Transcription Bottleneck in Quality Assurance 13
4. The Automation Gap: Failures of Existing Technologies 13
B. Definition of the Data Dichotomy in Engineering Workflows 13
C. Mechanical Drawings and Engineering Specifications 14
1. Drawing Standards and Regulatory Framework 15
2. The Vector vs. Raster Graphics Paradigm 16
3. The IGES Standard: Legacy but Reliable Data Source 17
4. Dimensional Specifications 18
5. Tolerance Systems 18
6. Geometric Dimensioning and Tolerancing (GD&T) 19
7. Visual Interpretation Challenges in Engineering Drawings 19
D. Problem Statement and Research Objectives 21
1. Problem Statement 21
2. Research objectives 22
Chapter II. Literature Review 24
A. Principle of Hybrid Intelligence 24
B. The "User-as-Selector" vs. "User-as-Validator" Design Pattern 25

C. Trust and Explainability in Industrial AI Systems 25

D. Mitigating AI Fallibility in Zero-Tolerance Environments. 26

E. Cognitive Load and Task Re-Allocation in Hybrid Systems 26

F. AI and Image Processing Techniques in Feature Extraction ... **Error! Bookmark not defined.**

1. Evolution of OCR and Text Recognition Methods 27

2. Modern Object Detection and Deep Learning Approaches 28

3. Prior Academic Research and Commercial Solutions 28

4. Architectural Considerations for Industrial Deployment 29

5. Template Matching: A Pragmatic Computer Vision Approach ..... **Error! Bookmark not defined.**

Chapter III. System Design ..... **Error! Bookmark not defined.**

A. Methodology Overview 31

B. Analysis of Existing Solutions and Applications ..... **Error! Bookmark not defined.**

C. Alternative Approaches and Rationale for Rejection 32

1. Generic OCR Methods 32

2. Pixel-Based Methods 33

3. End-to-End AI APIs 33

4. Third-Party APIs 33

5. Purely Vision-Based Object Detection 34

6. Machine Learning 34

D. System Architecture and Framework 35

E. Methodological Pivot: The IGES Data Extraction Breakthrough 36

F. The Multi-Modal Processing Pipeline 37

G. AI Integration for GD&T Classification 39

H. Automated Quality Report Generation and User Interaction: The Zero-Error Workflow 40

Chapter IV. Implementation Details 41

A. Development Environment and Tools 41

B. Graphical User Interface (GUI) Workflow 42

C. Implementation of the Processing Pipeline 47
1. IGES Parsing and Tokenization 47
2. PDF Span Extraction and Normalization 48
3. Multi-Modal Matching Algorithm 49
4. Computer Vision for Symbol Recognition 50
5. Unmatched Values and GD&T Frame Extraction 51
D. AI Integration for GD&T Classification 51
E. Automated Quality Report Generation 52
F. System Optimization and Error Handling 53
Chapter V. Results and Discussion 54
A. Case Study: Deployment at Ansaldo Energia 54
B. Performance Analysis 54
1. Quantitative Results 54
3. Comparative Analysis 55
C. Quality Report Generation Evaluation 55
D. Discussion 56
1. Interpretation of Findings 56
2. Implications for Industry 4.0 56
3. Limitations 56
Chapter VI. Conclusions and Future Work 57
A. Summary of Contributions 57
B. Conclusions and Commercial and Industrial Impact 57
C. Future Research Directions and Future Work 58
1. Cloud-Based SaaS Deployment ..... **Error! Bookmark not defined.**
2. Deep Enterprise Integration 58
3. Intelligent Selection Learning 58
4. Advanced Symbol Detection Models 59
5. Generalized Model Development: Cross-Industry Applicability 59
References 60

Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
ASME	American Society of Mechanical Engineers
CAD	Computer-Aided Design
CMM	Coordinate Measuring Machine
CV	Computer Vision
ERP	Enterprise Resource Planning
GD&T	Geometric Dimensioning and Tolerancing
GUI	Graphical User Interface
HITL	Human-in-the-Loop
IGES	Initial Graphics Exchange Specification
JSON	JavaScript Object Notation
LLM	Large Language Model
MBE	Model-Based Enterprise
MES	Manufacturing Execution System
ML	Machine Learning
NMS	Non-Maximum Suppression
OCR	Optical Character Recognition
PDF	Portable Document Format
PLM	Product Lifecycle Management
PMI	Product and Manufacturing Information
SaaS	Software-as-a-Service

General Introduction

The principles of Industry 4.0 are revolutionizing modern manufacturing, especially in high-stakes sectors like power generation where the precision of core components is non-negotiable. At the center of this ecosystem, quality assurance depends on mechanical engineering drawings, which serve as the official record for all design specifications. However, the manual interpretation of these documents—extracting dimensions, tolerances, and Geometric Dimensioning and Tolerancing (GD&T) symbols for inspection reports—remains a critical bottleneck. This process is not only labor-intensive and inefficient but also prone to human error, where small inaccuracies can escalate into significant production delays and financial losses.

This problem is amplified by the "data dichotomy" between machine-readable CAD files and the visually complex, unstructured PDF drawings that hold final authority. Traditional automated solutions, like generic Optical Character Recognition (OCR) or proprietary "black-box" APIs, are ill-equipped to handle the specialized symbols and formats, and their lack of transparency is unsuitable for regulated industries requiring verification. As a result, quality control processes remain disconnected from the integrated digital thread that defines a modern Model-Based Enterprise, hindering progress toward full automation.

This thesis confronts these challenges by designing, implementing, and validating a working software tool — a hybrid intelligence desktop application — for automated feature extraction from paired engineering drawings. The central output of this work is not a theoretical framework but a functional, deployable piece of software: a Python-based application with a graphical interface that an engineer can run, interact with, and use to produce a structured quality inspection report in minutes. The tool's core pipeline correlates tokenized geometric data from IGES files with PDF vector elements, augmented by targeted computer vision and a vision-language model, all governed by a mandatory human-in-command selection step that structurally prevents errors from reaching the output. This work demonstrates that such a tool is buildable, practically effective, and ready for integration into industrial quality assurance workflows.

Chapter I. Introduction

A. Introduction to Quality Assurance in Modern Manufacturing

1. The Industry 4.0 Imperative for High-Precision Manufacturing.

The fourth industrial revolution, or Industry 4.0, has fundamentally reshaped manufacturing by integrating digital technologies like automation, data analytics, and interconnected systems to create highly efficient production environments [1]. This paradigm emphasizes a shift towards data-driven manufacturing, where interconnected systems enable real-time decision-making and predictive maintenance.

In sectors such as power generation and heavy machinery, where components must be produced with high-level precision, quality assurance (QA) serves as the most critical function. It ensures that every part conforms to its design specifications, preventing defects that could lead to catastrophic failures or costly rework. Quality assurance challenges in power generation manufacturing are particularly acute, given the complexity of components and the high stakes involved in operational reliability [2].

2. The Engineering Drawing as the Primary Reference Document

Central to this entire process are mechanical engineering drawings, which act as the definitive legal and technical documents governing a component's lifecycle from design to inspection. These drawings establish the engineering blueprint as the cornerstone of product manufacturing, highlighting the criticality of accurate data extraction for downstream processes like machining, quality control, and assembly.

The fabrication of complex industrial components begins with Computer-Aided Design (CAD), where parts are modelled and simulated. These digital blueprints are then translated into manufacturing instructions via Computer-Aided Manufacturing (CAM) systems. However, the final, authoritative source of truth is not the CAD model itself, but the exported 2D engineering drawing, typically in Portable Document Format (PDF). This document encapsulates not only the component's geometry but also a dense layer of critical Product and Manufacturing Information (PMI). This includes dimensions, bilateral and unilateral tolerances, and complex Geometric Dimensioning and Tolerancing (GD&T) annotations, all governed by rigorous standards like ASME Y14.5 [3].

3. The Manual Transcription Bottleneck in Quality Assurance

These drawings serve as the primary interface for process engineers, who are tasked with transcribing this information to create inspection plans for tools like Coordinate Measuring Machines (CMMs). Historically, this has been an entirely manual process. Engineers would meticulously review drawings, identify critical features, and manually input the required nominal values and tolerance ranges into quality reports. The past struggles for process engineers included not only manual identification of dimensions and feature control frames but also manual input of values and manual editing of mechanical parts drawings.

This workflow is not only a significant operational bottleneck but is also fraught with the risk of human error. Even a minor transcription mistake in a tolerance value can lead to incorrect part acceptance or rejection, resulting in production halts and significant financial consequences. Industry 4.0 integration requirements demand non-invasive automation principles to address these manual annotation bottlenecks in engineering workflows.

4. The Automation Gap: Failures of Existing Technologies

The complexity of these drawings presents a formidable challenge for automation. Annotations are often densely packed, text may be rotated to various angles, and graphical symbols are used to convey precise geometric controls. Existing solutions have proven inadequate. Generic Optical Character Recognition (OCR) tools fail to reliably interpret specialized engineering fonts and symbols, while proprietary "black-box" commercial services present issues with cost, data privacy, and lack of transparency, making them unsuitable for industries where every step of the quality process must be verifiable.

Consequently, a major disconnect persists between the digital ambitions of Industry 4.0 and the realities of the QA workflow. While design and manufacturing have become increasingly digitized, the crucial link between the engineering drawing and the final inspection report remains a manual, inefficient, and error-prone bridge. This gap highlights an urgent need for an intelligent, transparent, and non-invasive automation solution that can seamlessly integrate into existing engineering processes and empower human expertise.

B. Definition of the Data Dichotomy in Engineering Workflows

The core challenge in automating the extraction of information from engineering drawings stems from a fundamental conflict referred to as the "Data Dichotomy." This term describes the split between two distinct forms of data that co-exist in the engineering workflow: the structured, machine-readable data within CAD files and the unstructured, visually contextualized data on the

authoritative PDF drawing. This disconnect creates a gap in the "digital thread"—the seamless flow of data from design through manufacturing to inspection—forcing manual intervention.

On one side, we have files like the Initial Graphics Exchange Specification (IGES). These are text-based formats that contain structured geometric and parametric data. Within these files, dimensional information—such as nominal values and tolerances—often exists as explicit, machine-readable text strings. However, this data is archaic in format and lacks the rich visual context of the final drawing. It does not visually link a dimension to its corresponding feature or include the graphical GD&T symbols that define complex geometric relationships.

On the other side is the PDF drawing. While modern PDFs created from CAD software are vector-based and contain embedded text information, the layout is designed for human interpretation, not machine parsing. Critical information is conveyed through the spatial arrangement of text, the use of graphical symbols (e.g., Ø for diameter, ⌖ for position), and the structure of feature control frames. This visual layer of information is unstructured and often eludes simple text extraction methods, representing the authoritative design intent that must be honored during inspection.

[INSERT FIGURE HERE — Caption: "Illustrative comparison of the same dimensional annotation as it appears in the structured IGES/CAD data versus its spatial rendering in the PDF drawing. Source: mock-up drawing." — Show side by side: left = raw IGES text token, right = the corresponding area on the PDF with the dimension visually placed among geometry.]

A key challenge is the hybrid nature of engineering data: while CAD files like IGES contain structured, machine-readable dimensional data, the definitive drawing—often a PDF—contains critical visual context, annotations, and GD&T symbols that may not be encoded in the CAD data. Because the IGES file has the raw numbers and the PDF has the essential context, a fully automated system cannot rely on one source alone. This forces engineers to perform manual cross-referencing between their transcriptions and the drawings, undermining the goals of a Model-Based Enterprise (MBE) and perpetuating a dependency on manual labor for a task that is fundamentally data-driven.

C. Mechanical Drawings and Engineering Specifications

Mechanical drawings represent the foundational communication medium in engineering and manufacturing, serving as the authoritative documentation that bridges conceptual design with physical reality. These technical documents encode not merely geometric information, but the complete manufacturing intent—encompassing dimensional specifications, tolerance requirements, surface finish criteria, and material properties. In the context of modern manufacturing, they function as legally binding contracts between design engineers, manufacturing personnel, and quality assurance

teams, establishing the precise parameters within which a component must be produced to fulfill its intended function.

1. Drawing Standards and Regulatory Framework

The standardization of mechanical drawing conventions is critical for ensuring global interoperability and eliminating ambiguity in manufacturing processes. The most prominent standard in North America is ASME Y14.5, the American Society of Mechanical Engineers' comprehensive specification for Geometric Dimensioning and Tolerancing (GD&T). This standard establishes a precise symbolic language for communicating allowable variations in part geometry, going beyond simple dimensional tolerances to address complex geometric relationships between features. ASME Y14.5 defines 14 fundamental GD&T symbols, each with specific mathematical definitions for form, orientation, location, and runout tolerances [6].

Additional critical standards include ISO 8015:2011 (Geometrical product specifications — Fundamentals: Concepts, principles and rules), which establishes the independence principle — the fundamental concept that each dimensional or geometric requirement specified on a drawing is met independently unless a relationship is explicitly stated. This is a key conceptual divergence from ASME Y14.5's envelope principle. Also part of the GPS family is ASME Y14.41 (Digital Product Definition Data Practices), which addresses the transition from traditional 2D drawings to model-based definition approaches [7].

In the European and Italian industrial context — directly relevant to this project's application environment — the ISO Geometrical Product Specifications (GPS) family of standards governs drawing conventions. The primary standard is ISO 1101:2017 (Geometrical tolerancing — Tolerances of form, orientation, location and run-out), which defines the symbolic language for geometrical specification and its interpretation rules, forming the GPS foundation [25]. General tolerance classes for dimensions and features not individually toleranced are specified in ISO 2768 (Parts 1 and 2), while ISO 8015 sets the overarching independence principle applied across all GPS standards. Together, these form the regulatory framework to which Italian manufacturers — including those in the power generation sector — conform, as opposed to or alongside the North American ASME Y14.5 system. Notably, both systems share the same GD&T symbol vocabulary (as standardized by ISO 1101), which is the primary target of this work's automated extraction pipeline [25].

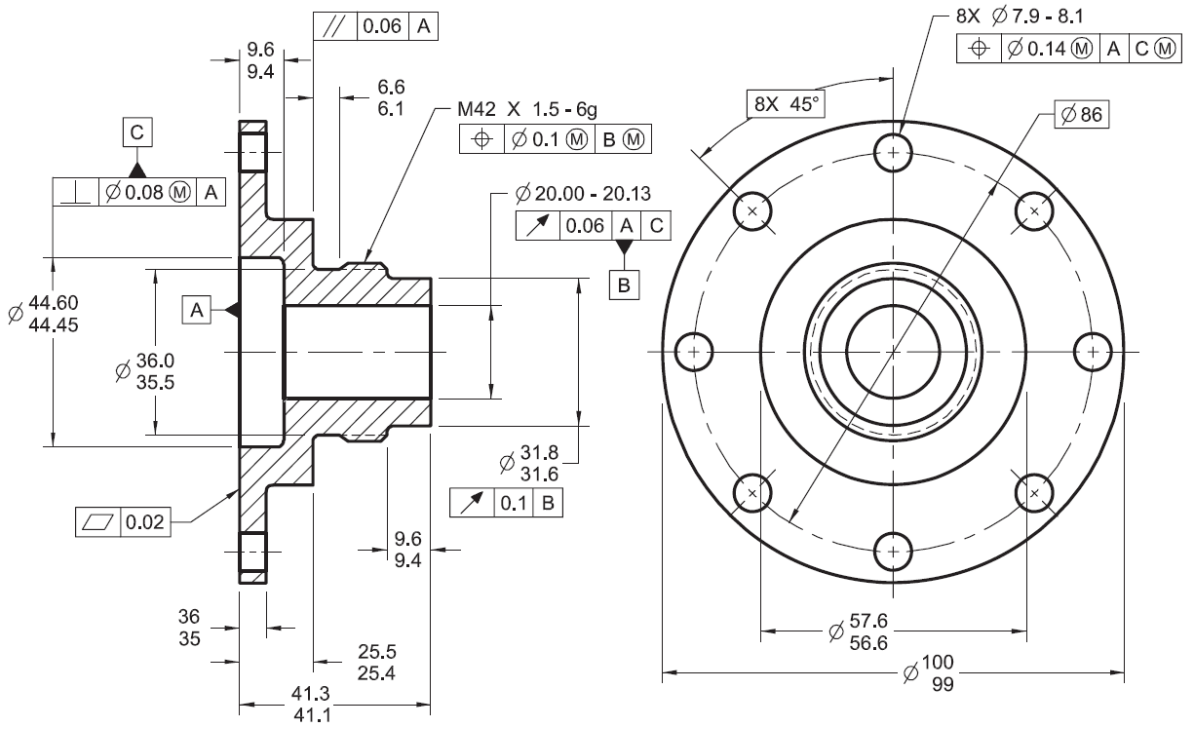

Figure I-1 Example of an engineering drawing

2. The Vector vs. Raster Graphics Paradigm

Understanding the fundamental distinction between vector and raster graphics is crucial for developing effective automated extraction systems. Vector graphics, predominant in modern PDFs generated directly from Computer-Aided Design (CAD) software, represent drawing elements as mathematical entities—lines are defined by coordinate pairs, arcs by center points and radii, and text by character codes with precise positioning data. This mathematical representation enables infinite scalability without quality degradation and, critically, embeds structured metadata including bounding box coordinates, rotation angles, and character encoding information.

This distinction fundamentally challenges the traditional approach of treating all drawing processing as an optical character recognition (OCR) problem. For modern vector-based PDFs, OCR is not merely inefficient—it is counterproductive, introducing unnecessary error potential where perfect accuracy is already available through direct vector data extraction. However, non-textual elements such as geometric symbols, feature control frames, and graphical annotations remain outside the vector text layer and require alternative detection methods.

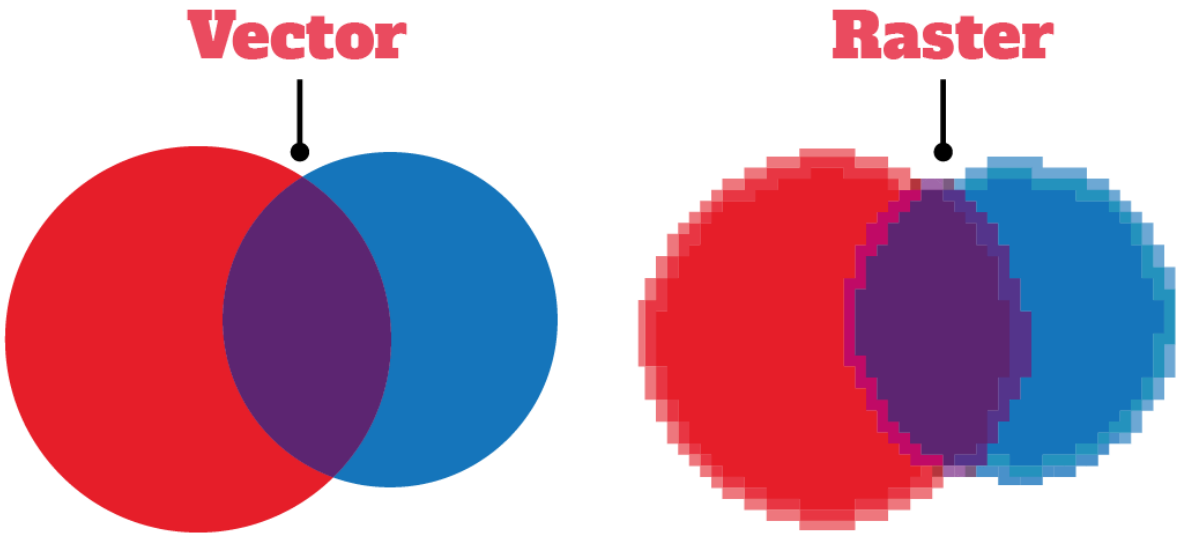

Figure I-1 https://reformcreative.co.uk/vector-vs-raster-files/ *[8]*

In contrast, raster graphics represent images as fixed grids of pixels, resulting in resolution-dependent quality and no embedded structural information. When drawings exist only as rasterized images, every textual element must be reconstructed through computer vision techniques, introducing potential errors and computational overhead.

The implications of this vector-based structure are profound for automated processing. When a PDF contains vector text, each character or string is stored with explicit spatial coordinates (x, y positions), orientation vectors (defining rotation angles), and font metrics. Libraries such as PyMuPDF (fitz) can directly access this structured data through APIs that return text spans with their associated bounding boxes (span['bbox']) and directional vectors (line['dir']), eliminating the need for optical character recognition entirely.

3. The IGES Standard: Legacy but Reliable Data Source

The Initial Graphics Exchange Specification (IGES), developed in the 1980s as a neutral file format for CAD data exchange, remains a valuable source of structured Product and Manufacturing Information (PMI) despite its archaic origins. IGES files organize data into parameter sections containing various entity types, often housing dimensional and tolerance information in structured text format [9].

While IGES lacks the geometric sophistication of modern standards like STEP (Standard for the Exchange of Product Data), its text-based parameter encoding makes it exceptionally suitable for

programmatic parsing. Dimensional annotations, tolerance specifications, and even basic GD&T information can be extracted using regular expressions and string parsing techniques, providing a "ground truth" dataset for correlating with visual elements in paired PDF drawings.

The reliability of IGES data stems from its direct generation from CAD systems during the design phase, before potential corruption or loss of information that can occur during PDF conversion or printing processes. This makes IGES files invaluable for establishing dimensional baselines against which PDF extractions can be validated.

4. Dimensional Specifications

Dimensions represent the nominal size specifications for geometric features and spatial relationships. These appear in various forms:

- **Linear dimensions**: Specify distances, lengths, widths, and heights
- **Angular dimensions**: Define rotational relationships between features
- **Radial dimensions**: Indicate radii of curved features, typically preceded by the radius symbol "R"
- **Diametral dimensions**: Specify diameters of cylindrical features, denoted by the diameter symbol "Ø"

5. Tolerance Systems

Tolerances define the permissible variation from nominal dimensions, ensuring parts function properly despite manufacturing imperfections. They manifest in several formats:

Symmetric tolerances (e.g., ±0.1) indicate equal positive and negative deviations from the nominal value. Asymmetric tolerances (e.g., +0.05/-0.02) specify different upper and lower bounds, often reflecting manufacturing process characteristics or functional requirements. Limit dimensions present maximum and minimum acceptable values directly (e.g., 25.0/24.8), eliminating the need for plus-minus notation.

Tolerance grades, such as H7, h6, or IT8, represent standardized tolerance classes based on fundamental deviation and tolerance magnitude. These alphanumeric codes encode both the direction of tolerance (uppercase letters for holes, lowercase for shafts) and the magnitude of permissible variation, enabling precise specification of fit relationships between mating parts.

6. Geometric Dimensioning and Tolerancing (GD&T)

Feature control frames are rectangular enclosures that contain comprehensive geometric tolerance specifications. These frames follow a standardized format: geometric characteristic symbol | tolerance value | material condition modifier | datum references. This systematic approach enables precise control of geometric relationships that cannot be adequately specified through dimensional tolerances alone.

Each symbol carries precise mathematical definitions for measurement and verification, establishing the foundation for coordinate measuring machine (CMM) programming and statistical process control.

Symbol	Characteristics	Categories
—	Straightness	Form
▱	Flatness	
○	Circularity	
⌭	Cylindricity	
∠	Angularity	Orientation
⊥	Perpendicularity	
//	Parallelism	
⌖	Position	Location
↗	Circular Runout	Runout
⌰	Total runout	
⌒	Profile of a line	Profile
⌓	Profile of a surface	

The 14 standard GD&T symbols defined in ASME Y14.5 comprise:

7. Visual Interpretation Challenges in Engineering Drawings

The complexity of visual interpretation in engineering drawings presents significant obstacles to automation. These documents represent compressed information spaces where multiple

annotation layers overlap, creating dense visual hierarchies that challenge both human interpreters and computer vision systems.

Scale and orientation variations further complicate extraction. Text and symbols may appear at any rotation angle, particularly in isometric or auxiliary views, requiring rotation-invariant recognition algorithms. Additionally, the same drawing may contain elements at different scales, demanding multi-resolution processing approaches.

Symbol ambiguity poses another critical challenge. Visually similar GD&T symbols—such as circularity (○) versus concentricity (⌖), or circular runout (↗) versus total runout (⌰)—require high-resolution analysis and contextual understanding to differentiate accurately. The distinction between these symbols carries significant manufacturing and inspection implications, making misclassification costly.

Annotation density creates scenarios where dimensional callouts, tolerance specifications, geometric symbols, and textual notes compete for limited drawing space. This results in overlapping elements, rotated text orientations, and reduced character sizes that complicate automated recognition. Stacked tolerances exemplify this challenge, requiring algorithmic parsing of hierarchical notation where multiple tolerance values apply to different aspects of a single feature.

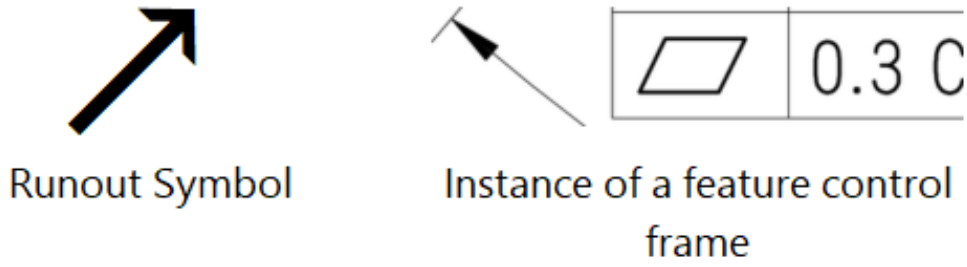

Figure I-1 Example of potential misidentifications

Figure I-2 Examples of Visually Similar Characters and GD&T Symbols

D. Problem Statement and Research Objectives

1. Problem Statement

Within the quality assurance workflow for high-precision manufacturing, the manual extraction of dimensions, tolerances, and GD&T from 2D engineering drawings represents a major source of inefficiency and risk. This labor-intensive task creates a significant bottleneck, delaying the generation of inspection reports and increasing product lead times. More critically, the process is susceptible to human error, where incorrect data entry can lead to the approval of faulty components or the rejection of conforming ones, incurring substantial financial costs.

The core problem is identified as the profound inefficiency and unreliability of manual data transcription from engineering drawings. This represents a significant bottleneck in modern manufacturing workflows, being not only labor-intensive but also a frequent source of costly human errors. The practical and financial limitations of existing commercial solutions further exacerbate this issue, underscoring the need for user-friendly, seamlessly integrated automation tools designed specifically for process engineers.

To address this challenge, there is a clear need for an automated solution that can:

- Improve efficiency and reduce lead times
- Enhance data accuracy and consistency
- Create a structured digital thread from design to inspection

However, existing technologies are insufficient. General-purpose OCR systems and commercial "black-box" APIs fail to provide the required accuracy and transparency for this specialized domain. They often misinterpret critical symbols and offer no mechanism for user validation, making them unacceptable in a quality-critical environment where every reported value must be verifiable. This leaves a critical gap for a tool that is precise, verifiable, and designed to integrate smoothly into the process engineer's existing workflow.

This research proposes the development and implementation of a hybrid intelligence system designed to automate the generation of quality reports from paired IGES and PDF drawings. By leveraging a multi-modal approach that correlates deterministic data from IGES files with vector information on PDFs, augmented by targeted computer vision, a mandatory human-in-command validation loop, and optional suggestions from LLM API calls, the system aims to eliminate transcription errors entirely.

The developed system addresses these challenges through a "glass-box" multi-stage pipeline, the methodology of which is defined in detail in Chapter III following the literature review.

1.
2.

2. Research objectives

- The overarching goal of this thesis is to develop a working software tool — a functional desktop application — that automates the generation of quality inspection reports from paired IGES and PDF engineering drawings. The three supporting technical objectives are:
- Primary Objective: To implement, within the tool, reliable detection of dimensions, bilateral/unilateral tolerances, tolerance grades, and associated symbols from paired IGES and PDF drawings; and to isolate and extract GD&T feature control frames for classification using a prompted Vision-Language Model.
- Secondary Objective: To integrate these detection capabilities into a complete, interactive desktop application with a human-in-command validation loop, capable of producing structured Excel quality reports compatible with Industry 4.0 frameworks. Effectiveness to be demonstrated through an industrial case study.

Tertiary Objective: To demonstrate, through the implemented tool, a transparent and cost-effective "glass-box" architecture that is functionally superior to opaque, recurring-cost commercial APIs.

a. Scope and Limitations

The project focuses on 2D engineering drawings in PDF format, paired with corresponding IGES files as the source of "ground truth" for dimensions. The system is designed to handle text at various angles and scales, with the user interface developed as a desktop application using web technologies.

b. Scope is limited to:

- 2D engineering drawings provided as vector or high-quality raster PDFs
- Corresponding IGES files for dimensional ground truth
- Desktop application deployment using web technologies

This thesis presents a novel Hybrid Intelligence-by-Design workflow, not merely an "AI tool", but a complete, end-to-end system where deterministic extraction, targeted machine vision, and mandatory human-in-the-loop selection structurally prevent unverified data from entering the final output.

Primary limitations include:

- Classification of complex GD&T symbols relying on a cloud-based LLM API (Google Gemini)
- Performance varying based on input PDF quality (e.g., raster scans vs. vector-based drawings)
- Considerations of cost, latency, and data privacy associated with API dependency

Validation approach:

The system will be validated through a case study using a diverse set of 10 technical drawings, demonstrating structurally enforced output correctness and seamless workflow integration in a live industrial environment.

To protect Ansaldo Energia's intellectual property, a simple demo engineering drawing was used in all illustrative examples and testing demonstrations, ensuring no proprietary data was exposed.

c. Target Feature Extraction

Examples of target feature extraction capabilities include:

- Dimensional tolerances (e.g., ±0.1, +0.05/-0.02)
- GD&T symbols
- Hole patterns and repetitive features

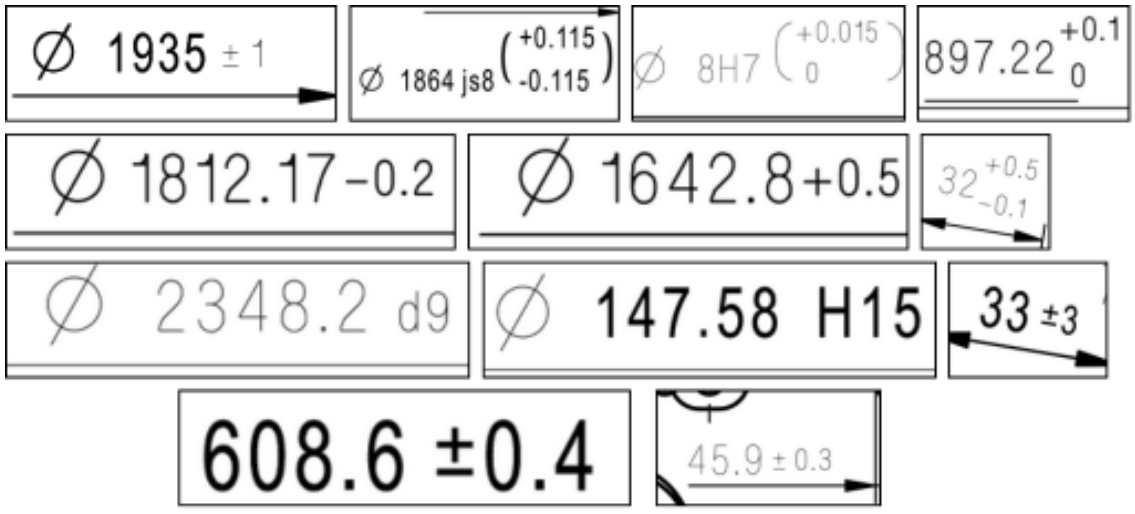

Figure I-3 Examples of features necessitating extraction

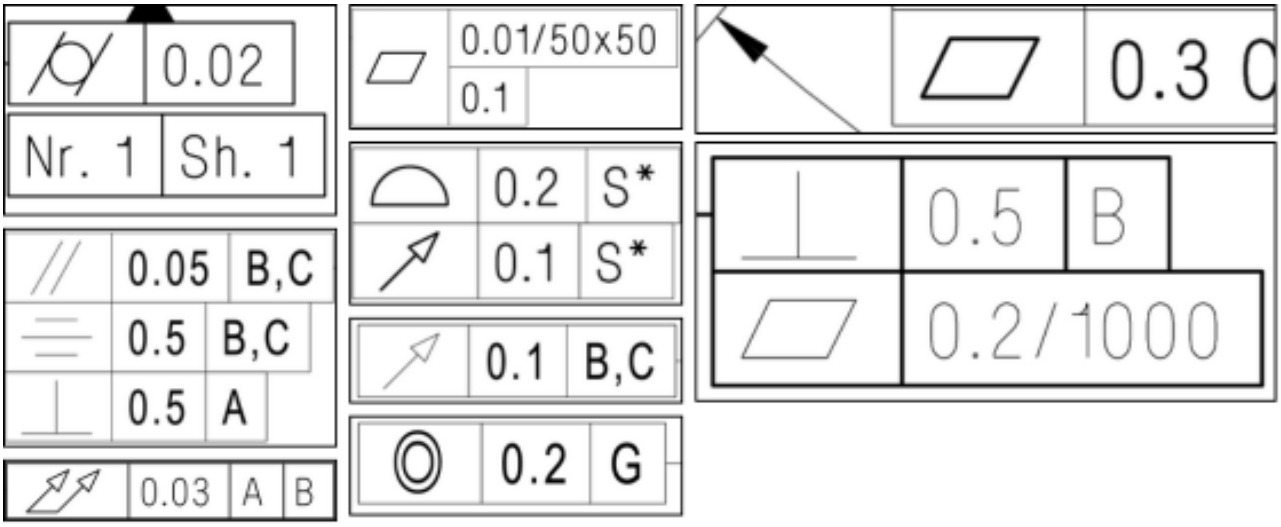

Figure I-4 Examples of other features necessitating detection

These visual examples demonstrate the complexity and variety of annotations that previously required manual identification, value input, and drawing modification by process engineers, highlighting the significant potential for automation in this domain.

Chapter II. Literature Review

A. Principle of Hybrid Intelligence

Hybrid intelligence represents a paradigm in AI systems that combines human cognitive capabilities with machine learning and computational strengths to achieve outcomes superior to those possible by either alone. This approach is particularly valuable in domains requiring high precision and accountability, such as quality assurance in manufacturing, where automated tools must augment rather than supplant human judgment and domain expertise [4].

A key distinction in hybrid intelligence frameworks is between "Human-in-the-Loop" (HITL) and "Human-over-the-Loop" models. In HITL systems, humans are actively involved in the core decision-making process, providing feedback, corrections, or validations at critical stages to refine AI outputs. This ensures that the system benefits from human expertise in handling ambiguous or context-dependent scenarios that AI might misinterpret. Conversely, "Human-over-the-Loop" models position humans in a supervisory role, overseeing automated processes but intervening only in exceptional cases, which can lead to over-reliance on AI and potential errors if the machine's suggestions are flawed.

The "user-as-selector" model proposed in this thesis advances a more robust "Human-in-Command" paradigm specifically tailored to the quality assurance workflow in manufacturing environments. In this framework, AI serves as an intelligent detection engine that identifies and extracts all available dimensions, tolerances, and symbols from engineering drawings with designed-for-purpose accuracy. However, the human operator—typically a process engineer—retains absolute authority over determining which detected elements are relevant for the specific inspection requirements and should be passed to mechanical technicians for measurement.

B. The "User-as-Selector" vs. "User-as-Validator" Design Pattern

This distinction is critical. Much of the existing literature focuses on a "user-as-validator" model, where the AI makes a suggestion, and the human's role is to check it for errors. This pattern, however, is highly susceptible to automation bias—a well-documented phenomenon where human operators become complacent and over-trust automated systems, leading them to miss subtle but critical errors. In this model, the human is merely a fallible safety net for an imperfect AI. In contrast, the "user-as-selector" model reframes this relationship entirely. The AI's task is not to suggest an inspection plan but to detect all possible features comprehensively. The human operator's task is elevated from error-checking to the high-level cognitive work of defining the inspection scope based on their domain expertise. This fundamentally re-architects the workflow to mitigate bias and places the human in a position of strategic command, not reactive validation. This architectural decision reflects the industrial reality that not all dimensions present on an engineering drawing require measurement during quality inspection. Process engineers possess the domain expertise to understand which features are critical for the specific manufacturing context, customer requirements, and quality standards. By positioning the system as a comprehensive detection tool rather than a selective filter, the human operator can make informed decisions about inspection scope based on complete information rather than pre-filtered suggestions.

The system is engineered for highly reliable deterministic detection of dimensional data through IGES parsing and precise correlation with PDF vector text, ensuring that critical dimensional information is recovered from structured sources rather than inferred probabilistically. The only component that employs probabilistic machine learning—specifically Large Language Models (LLMs)—is reserved exclusively for the classification of Geometric Dimensioning and Tolerancing (GD&T) feature control frames, where the visual complexity and symbolic nature of the annotations necessitate advanced pattern recognition capabilities.

C. Trust and Explainability in Industrial AI Systems

This deliberate architectural choice directly addresses the challenge of trust and adoption in industrial settings. Opaque, "black-box" AI systems often face resistance in high-stakes environments due to their unpredictable failure modes, lack of auditability, and concerns over data privacy when using third-party APIs. Literature in Human-Computer Interaction (HCI)

emphasizes that trust is fostered through transparency and predictability [22]. Our system's "glass-box" design, which is founded on a deterministic and therefore explainable core, provides this transparency. By explicitly isolating the probabilistic LLM to a single, non-critical classification task that is subject to mandatory human review, the system contains the risks of AI fallibility and builds a foundation of trust with the user.

This mandatory human-driven selection workflow structurally prevents errors in the final quality reports: because only user-confirmed elements are included, no unverified data can reach the output. Dimensional accuracy in the detection phase is maintained independently of this selection step. The user's role is re-architected from a fallible validator of AI detections to an infallible selector of inspection scope, eliminating the possibility of measurement errors due to missed or incorrectly interpreted dimensional specifications.

D. Mitigating AI Fallibility in Zero-Tolerance Environments.

The necessity for a Human-in-Command model is underscored by the zero-tolerance nature of manufacturing quality assurance. In this domain, a single misinterpreted tolerance can result in significant financial loss through scrap, rework, or even catastrophic product failure [23]. The typical 80-90% accuracy rates celebrated in general AI research are wholly unacceptable here. Our methodology acknowledges the inherent fallibility of probabilistic models like OCR or LLMs, which are prone to "hallucinations" or subtle misclassifications. Rather than attempting to achieve perfect autonomy, our system responsibly contains these fallible technologies. It uses deterministic methods for all critical dimensional data and deploys the LLM only for classifying symbolic information that an expert subsequently verifies, thus ensuring AI fallibility is managed and cannot propagate into the final, critical output.

By design, this approach mitigates risks associated with AI hallucinations or inaccuracies in critical dimensional data while leveraging machine intelligence where it provides clear value—in comprehensive detection and preliminary classification of complex visual symbols. This alignment with ethical and practical demands in high-stakes industrial environments ensures that the system enhances rather than replaces human expertise, supporting the broader goals of Industry 4.0 digital transformation while maintaining the reliability standards essential for manufacturing quality assurance.

E. Cognitive Load and Task Re-Allocation in Hybrid Systems

One of the core principles of effective hybrid intelligence is the intelligent re-allocation of cognitive tasks. The goal is not simply to automate work, but to offload the repetitive, tedious, and error-prone aspects of a task, thereby freeing the human expert to focus on higher-value cognitive

functions. In this case, the system automates the laborious and mentally taxing process of manually locating, transcribing, and correlating data from disparate sources. This does not diminish the engineer's role; it elevates it. By removing the cognitive load of data entry, the system empowers the engineer to dedicate their full expertise to interpreting design intent, assessing manufacturing priorities, and making strategic decisions about quality control—tasks where human judgment remains irreplaceable.

Literature on hybrid intelligence, such as works by Dellermann et al. [5] on socio-technical systems, emphasizes that such human-centric designs enhance trust, adoption, and overall system efficacy in complex tasks where domain expertise and contextual judgment remain irreplaceable components of the decision-making process.

F. F. AI and Image Processing Techniques in Feature Extraction

The application of artificial intelligence and computer vision to engineering drawing analysis has evolved through multiple technological generations, each addressing specific limitations of predecessor approaches while introducing new challenges and capabilities.

1. Evolution of OCR and Text Recognition Methods

Traditional Optical Character Recognition (OCR) techniques, exemplified by systems like Tesseract and EasyOCR, operate on the fundamental principle of pattern matching between image regions and trained character templates. These systems excel in document processing applications with standard fonts and clean backgrounds but encounter significant difficulties with engineering drawings' specialized requirements [11].

Engineering-specific challenges include font variation tolerance, where technical drawings employ fonts optimized for clarity at small scales, often with non-standard character spacing and stroke weights. Symbol disambiguation represents another critical limitation—OCR systems frequently misidentify specialized engineering symbols such as the diameter symbol (Ø) as the letter 'O' or the digit '0', or plus-minus symbols (±) as separate '+' and '-' characters.

Spatial relationship parsing poses additional complexity, as OCR systems typically focus on character recognition without understanding the geometric relationships between text elements that define dimensional and tolerance specifications. The critical association between a dimension value and its corresponding tolerance notation requires spatial analysis beyond standard OCR capabilities.

2. Modern Object Detection and Deep Learning Approaches

Contemporary object detection models, including YOLO (You Only Look Once), Faster R-CNN, and transformer-based architectures, offer potential advantages for identifying annotation bounding boxes and classifying drawing elements. These models can theoretically handle multiple object classes simultaneously while providing spatial localization information essential for drawing analysis [12].

However, engineering drawing applications present unique challenges that limit the effectiveness of generic object detection frameworks. Training data scarcity represents a primary obstacle, as publicly available datasets for engineering drawing elements are limited, and proprietary industrial drawings cannot be shared for model training due to intellectual property constraints. Symbol variability compounds this challenge, as the same geometric tolerance symbol may appear with different visual characteristics depending on drawing scale, CAD software version, and export settings [11] [13].

Computational resource requirements for deep learning approaches often exceed the constraints of industrial deployment environments, where quality assurance systems must operate on standard office computers without specialized GPU hardware

3. Prior Academic Research and Commercial Solutions

Academic research in engineering drawing analysis has produced promising results in controlled environments but often lacks the robustness required for industrial implementation. Studies such as Moreno et al. [11] demonstrate high accuracy rates for isolated symbol recognition tasks but encounter significant performance degradation when applied to complete drawings with realistic complexity levels.

Several commercial solutions have emerged for automated engineering drawing analysis. Werk24 (werk24.ai) is a publicly available API service specifically designed for mechanical drawing feature extraction: it accepts PDF drawings and returns structured data including dimensions, tolerances, and GD&T annotations via a REST API. While Werk24 demonstrates the commercial demand for this capability, it operates as a closed, opaque system — offering no visibility into its extraction logic, no mechanism for mandatory user validation before output is accepted, and usage-based pricing that scales poorly for high-volume industrial applications. Similar constraints apply to general-purpose document AI APIs (such as Google Document AI or AWS Textract) when repurposed for engineering drawings: they lack domain-specific symbol vocabularies, cannot reliably distinguish GD&T feature control frames from surrounding geometry, and provide no structured human-in-the-loop confirmation step. The fundamental limitation shared by all these commercial offerings is architectural: by treating extraction as a fully

automated endpoint rather than a human-assisted workflow, they are unsuitable for zero-error-tolerance quality assurance environments where every output must be verifiable.

4. Architectural Considerations for Industrial Deployment

The selection of web-based architecture using lightweight backend frameworks (Flask) combined with client-side JavaScript reflects practical deployment constraints in industrial environments. This approach minimizes infrastructure requirements, simplifies installation and maintenance procedures, and provides cross-platform compatibility essential for diverse manufacturing IT ecosystems.

Asynchronous task handling through client-server architectures enables non-blocking user interfaces that maintain responsiveness during computationally intensive analysis operations. Background processing using Python's multiprocessing capabilities allows complex drawing analysis to proceed without freezing user interfaces, improving operator acceptance and workflow integration.

5. Template Matching: A Pragmatic Computer Vision Approach [24]

Template matching techniques, implemented through OpenCV's correlation-based matching algorithms, provide a computationally efficient alternative to complex machine learning approaches for detecting specific geometric symbols and annotations. This classical computer vision technique compares image regions against predefined templates, identifying locations where correlation exceeds specified thresholds.

Template matching offers several practical advantages for engineering drawing applications:

- Computational efficiency: it requires minimal processing resources compared to neural network approaches, making it suitable for desktop deployment without GPU infrastructure.

- Interpretability: matched locations correspond directly and transparently to predefined template positions, with no learned feature abstractions that require explanation.

- No training data required: unlike deep learning methods, template matching is immediately applicable to specialized industrial domains where large, annotated datasets are unavailable or impractical to assemble.

Multi-scale template matching addresses scale variation challenges by evaluating templates at multiple size ratios, ensuring detection robustness across different drawing scales and zoom levels.

Rotation handling can be implemented through template rotation or image normalization techniques, though the computational cost increases with the number of angular variations considered.

The integration of template matching within a larger processing pipeline—where deterministic text extraction and heuristic spatial analysis define targeted search regions—demonstrates a hybrid approach that leverages the strengths of multiple techniques while mitigating individual limitations.

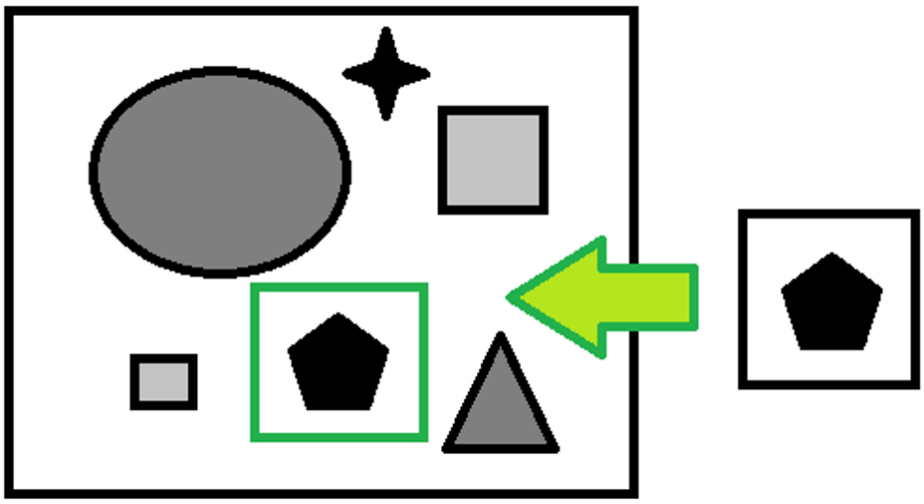

Figure II-1 Template Matching Illustration [14]

Chapter III. System Design

A. Methodology Overview

The system is built on a "glass-box" methodology that ensures the final output is user-verified. Unlike opaque automated pipelines, every stage of processing is inspectable and every final decision is made by the human operator. The multi-stage pipeline consists of:

Deterministic vector-text and IGES parsing for dimensional ground truth
Heuristic-based tolerance matching correlating structured and spatial data
A mandatory user-driven selection and finalization loop that prevents any unverified element from entering the output

This methodology is defined here, after the literature review has established the landscape of available techniques, to demonstrate that the design choices are grounded in the state of the art and not arbitrary. The following sections justify each component of this architecture by comparing it to rejected alternatives before presenting the full system design.

B. Analysis of Existing Solutions and Applications

The automation of feature extraction from engineering drawings has been a persistent challenge, with existing methodologies falling into several categories, each with inherent limitations that render them unsuitable for high-stakes quality assurance in manufacturing.

A common approach begins with Optical Character Recognition (OCR), treating the drawing as an image to be scanned for text. However, this is fundamentally inefficient for modern vector-based PDFs, which already contain structured, machine-readable text that can be extracted deterministically without the error-prone approximations of OCR. Tools like those relying on pixel-level analysis often misidentify specialized symbols, such as confusing 'Q' for 'Ø' or '+' for '±', due to variations in engineering fonts and graphical representations. These inaccuracies make OCR-based methods impractical for environments demanding perfect fidelity, as even minor errors can propagate into costly manufacturing defects.

Commercial solutions, such as API services for drawing analysis, offer automated extraction but come with operational drawbacks. Their cost structures, often based on per-use fees, can become prohibitive for high-volume industrial applications. Additionally, reliance on external services raises concerns about data privacy, as proprietary drawings must be transmitted over networks, potentially exposing intellectual property in regulated sectors. Compliance with industry standards for data handling further complicates adoption. Critically, these solutions typically lack an integrated layer for human selection or validation, limiting their utility in workflows that require guaranteed accuracy through user oversight.

Academic research has advanced detection techniques but often within constrained scopes. Studies achieving high accuracy typically focus on a narrow set of symbols or simplified drawings, failing to scale to the diverse, complex annotations found in real industrial documents. Reported accuracy rates of 80-90% are noteworthy for exploratory purposes but unacceptable in quality assurance, where a single misdetection could lead to significant financial, safety implications or an interruption of a supposedly semi-automated workflow. Moreover, much of the literature emphasizes isolated extraction algorithms without addressing end-to-end integration, including user interfaces for scope selection and report generation—elements essential for practical deployment.

Overall, the state of the art reveals a gap: while tools exist for partial automation, none provide a comprehensive, verifiable system that combines precision extraction with mandatory human command, ensuring seamless fit into existing manufacturing workflows.

C. Alternative Approaches and Rationale for Rejection

The development process was iterative, involving experimentation with various techniques before arriving at the final methodology. This section details the discarded approaches, providing rationale for their rejection based on empirical failures, scalability issues, and misalignment with project requirements for accuracy, transparency, cost and integration.

1. Generic OCR Methods

An evaluation of generic OCR libraries, including Tesseract (an open-source engine by Google for image text recognition), EasyOCR (a Python package for multi-language detection), and eDocr (a specialized document tool from preliminary studies), applied them to rasterized drawings for text and symbol extraction.

They failed consistently due to engineering fonts' stylized elements not in standard datasets. For example, 'Ø' was often misread as 'O' or '0', and '±' as separate signs, corrupting data. Tools also struggled with rotated text and dense annotations, losing spatial relationships needed for dimension-tolerance associations and fragmenting outputs. Inability to parse structures like stacked tolerances made them unsuitable for absolute accuracy in quality assurance, leading to rejection for vector-based methods using embedded PDF data.

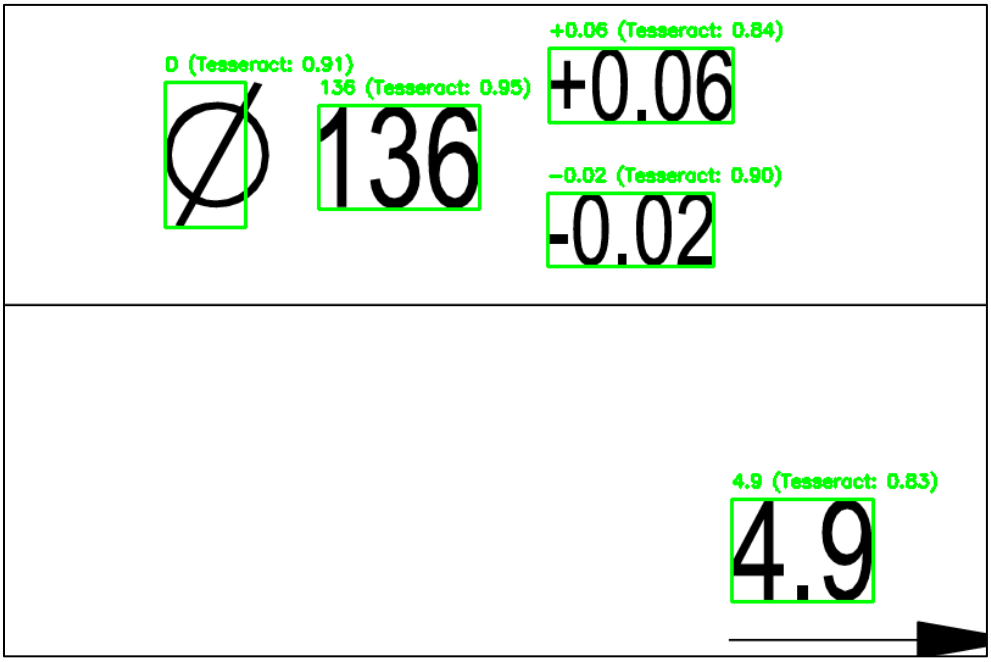

Figure III-1 Example of attempted OCR using Tesseract

2. Pixel-Based Methods

Early attempts used pixel-based template matching with OCR for symbol and text detection. Template matching compares image patches to predefined patterns, but trials showed limitations with rotations, scales, or leader lines causing mismatches and false results. Integrated OCR retained low accuracy on specialized fonts, like confusing 'Q' and 'Ø'.

The method was brittle against scan noise or artifacts, and scaling to full GD&T sets required exhaustive templates, undermining automation. Rejected for unreliability and inefficiency, favoring targeted vector-informed regions.

3. End-to-End AI APIs

Commercial AI APIs were assessed for direct extraction of dimensions and tolerances. Capable of structured outputs, they were rejected for high recurring costs in industrial volumes, restrictive licensing limiting customization, and security risks from transmitting proprietary drawings, potentially breaching IP in regulated sectors.

They lacked human-selector integration for validation, essential for structurally enforced output correctness, and testing revealed GD&T inconsistencies, necessitating an in-house transparent alternative.

4. Third-Party APIs

Broader third-party APIs for vision and text processing were considered but abandoned. Beyond costs and licensing, the primary issues were data privacy risks from external transmission and the inability to embed a mandatory validation loop. Without control over the underlying models, adapting to specific drawing styles or standards was challenging, leading to rejection in favor of a bespoke system.

5. Purely Vision-Based Object Detection

Purely vision-based methods, aiming to treat the drawing as an image for end-to-end detection, were explored extensively.

a. Template Matching

Initial success with symbols like 'Ø' and '±' failed to scale. Variations in line thickness, arrows, or elements (e.g., single vs. double in concentricity) caused high error rates. Rendering inconsistencies across CAD made universal templates impractical, with full-page scans computationally heavy.

b. Contour Detection & Grouping

Computer vision detected primitives (lines, arcs, circles, rectangles), but drawings' thousands of elements created combinatorial explosions in clustering. Attempts to form symbols (e.g., position from circle/crosshairs, or feature control frames from rectangles) were brittle to incomplete contours or overlaps, proving intractable for production use.

c. Contour & Shape Detection:

Contour-finding highlighted unreliability: frames often broken or merged with geometry, resulting in misdetections in dense areas. Variations across standards amplified issues, rejecting for lack of robustness in favor of structured data guidance.

6. Machine Learning

Supervised models (e.g., via TensorFlow/PyTorch convolutional networks) were explored for feature recognition from images. Prototypes fine-tuned pre-trained models on annotated datasets.

Discarded due to challenges: dataset collection was tedious, needing thousands of diverse labels with bounding boxes, demanding expertise and time beyond resources. Public datasets lacked industrial specificity, causing poor generalization. Models achieved 80-90% accuracy—

insufficient for quality risks—and confused similar symbols or dense regions, requiring heavy post-processing. Training/inference demands and decision opacity clashed with "glass-box" needs, favoring deterministic heuristics without data training.

D. System Architecture and Framework

The system's architecture is designed as a multi-stage data processing pipeline that transforms unstructured and semi-structured inputs—PDF drawings and IGES files—into a fully structured, verified output in the form of an Excel quality report. This high-level design emphasizes modularity, with distinct stages for ingestion, matching, detection, and user interaction, orchestrated by a Python backend. The user interacts through a graphical interface built with web technologies, ensuring accessibility and ease of use in a desktop environment.

The overall system flowchart illustrates the flow: starting with file upload (PDF and IGES), proceeding to processing triggers, JSON data updates, and UI refreshes. A client-server model handles asynchronous tasks, allowing background processing without interrupting the user experience. The feature extraction pipeline is detailed as follows:

- Stage 1: Parallel Data Ingestion – The system parses the IGES file to extract tokenized dimensional data while simultaneously using a PDF library to retrieve all vector text spans from the drawing.
- Stage 2: Deterministic Matching – IGES tokens are heuristically matched to PDF text spans based on proximity and reading order.
- Stage 3: Unmatched Value Identification – The code filters for standalone numerical texts likely to be feature control frames, defining dynamic search areas and creating exclusion zones around matched tolerances to optimize processing.
- Stage 4: Targeted Symbol Detection – Template matching is applied within cropped search areas to identify graphical symbols.
- Stage 5: Data Persistence – All extracted data, including matches and unmatched conditions with associated image snippets, is stored in a JSON file serving as the central data source.

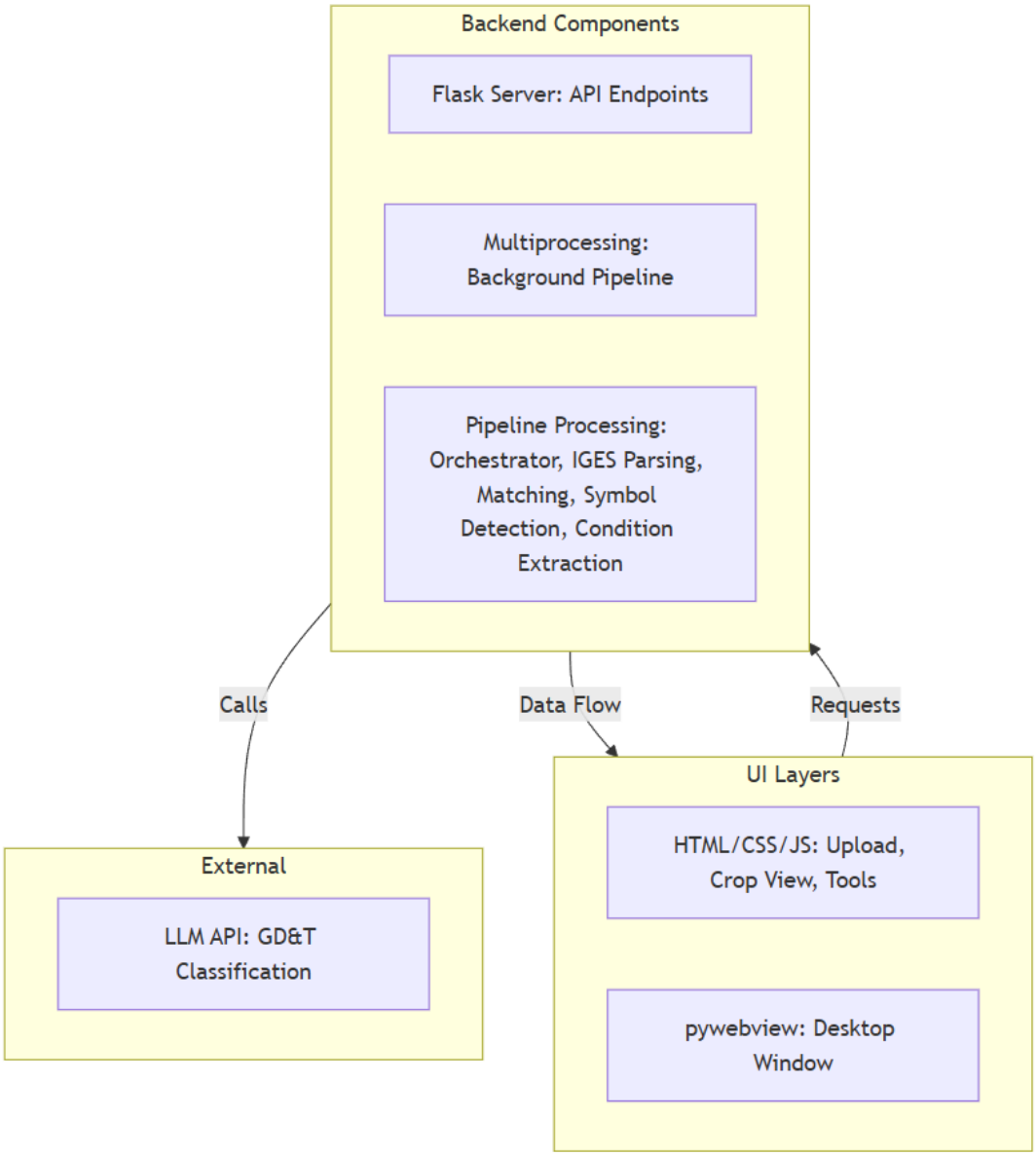

Figure III-1 Detailed System Flowchart Illustrating the Multi-Stage Pipeline

E. Methodological Pivot: The IGES Data Extraction Breakthrough

The project's methodology underwent a significant evolution, shifting from an initial focus on pure computer vision to a more robust, data-driven approach. Early efforts treated the problem as one of visual recognition, attempting to detect and interpret all elements directly from the PDF drawing. However, this proved challenging due to the variability in symbol rendering, text orientations, and overlapping annotations, leading to inconsistent results.

The breakthrough came with the realization that much of the required dimensional and tolerance data was already embedded in a structured, text-based format within the IGES file's general note entities. This discovery reframed the core challenge: instead of recognizing unknown text and symbols from scratch, the task became one of correlating known tokens from the IGES with their spatial counterparts in the PDF. By leveraging the IGES as a reliable source of "ground truth" tokens (e.g., sequences like 'Ø', a numerical value, and a tolerance grade), the system could focus on heuristic-based location and grouping in the drawing's vector layer. This pivot enabled higher accuracy and efficiency, forming the backbone of the final architecture and allowing targeted vision techniques to handle only the remaining graphical elements.

F. The Multi-Modal Processing Pipeline

The processing pipeline integrates multiple modalities—textual, spatial, and visual—to extract and correlate features from paired IGES and PDF files.

The pipeline comprises five sequential stages, each illustrated in the figures below.

- **Stage 1: IGES Parsing and Tokenization**

The code reads IGES files, focusing on text-based entities to identify dimensional annotations. Regular expressions deconstruct raw strings into structured tokens, such as ordered lists (e.g., ['Ø', '1864', 'H7']), which serve as search queries for matching in the PDF. This rule-based extraction is efficient and reliable for the structured text format, outperforming machine learning alternatives in speed and precision for this domain.

- **Stage 2: PDF Span Extraction and Normalization**

Using a PDF parsing library, the code extracts granular text spans, each with content, bounding box, and orientation vector. The orientation is converted to degrees, and Unicode variations of symbols (e.g., different encodings of 'Ø') are standardized to ensure consistency in matching.

- **Stage 3: Multi-Modal Matching Algorithm**

At the pipeline's core, the algorithm generates candidate groupings of PDF text spans that match IGES token content using combinatorial methods. Validations ensure spatial compactness and orientation consistency, preventing erroneous associations of distant elements. A composite scoring metric rewards tight bounding boxes and semantic reading order, selecting the most plausible match based on geometric and syntactic criteria. Heuristics, such as proximity thresholds and order calculations, handle rotated text effectively.

- **Stage 4: Computer Vision for Symbol Recognition**

For graphical symbols absent from the PDF text layer (e.g., Ø or ±), the code invokes targeted vision processing. It defines search regions relative to known text, renders them as high-resolution images, normalizes orientation via affine transformations, and applies multi-scale template matching. Non-Maximum Suppression refines detections to the highest-confidence result. Optimizations, like downsampling, balance precision and speed for responsive performance.

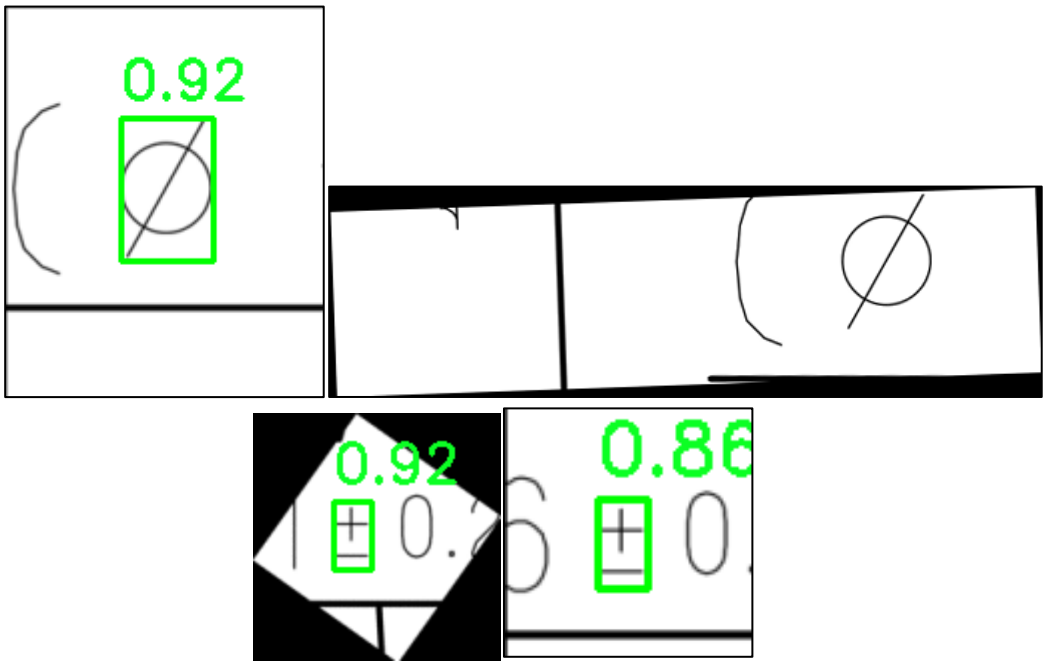

Figure III-2 Template Matching with NMS example

- **Stage 5: Unmatched Condition and GD&T Frame Extraction**

The code identifies standalone numerical conditions (e.g., '0.1') as part of GD&T frames, calculating oriented search areas to their left where symbols are expected. Dynamic heuristics adjust search box sizes based on text length, using vector math for rotation handling. The areas are rendered as aligned image snippets, saved for classification, with exclusion zones preventing redundant processing of matched regions.

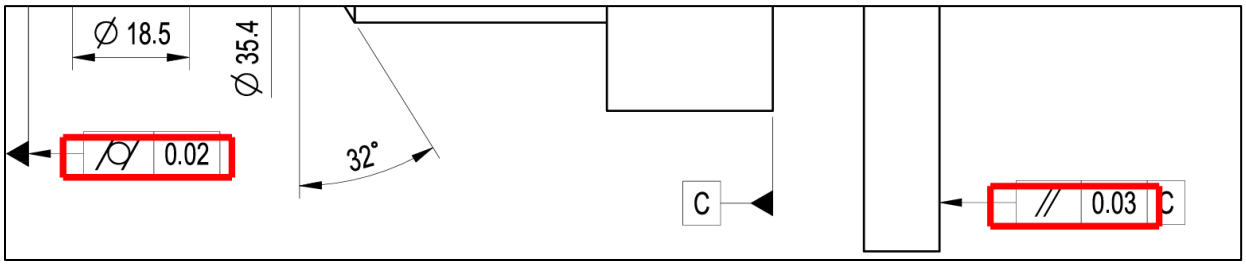

Figure III-3 Illustration of Dynamic Search Area Calculation

G. AI Integration for GD&T Classification

Prompt engineering involves crafting structured inputs to elicit precise, reliable outputs from Large Language Models (LLMs). For vision-language models, prompts combine text instructions with images, guiding analysis in multi-modal contexts . In this system, image snippets from unmatched values are classified via a cloud-based LLM API.

Prompt Engineering for Vision-Language Models

The prompt is engineered for classifying GD&T symbols in snippets, broken into components:

- The prompt specifies identifying the leftmost symbol and checking for modifiers like "Ø".
- The prompt supplies a dictionary of the 14 GD&T symbols with visual descriptions (e.g., position as a circle with crosshairs).
- A "Key Distinctions" section highlights differences between similar symbols (e.g., single vs. double circles for circularity and concentricity), improving accuracy by addressing common confusions.
- Step-by-step reasoning is encouraged to support logical analysis.
- The response format is constrained to a parsable structure such as "[Symbol Name] | Modifier 'Ø' present: [Yes/No]" or "Unclear" for integration.

The template ensures clarity by listing distinguishing features (e.g., tangent lines for cylindricity), resolves ambiguities, mandates structured responses, and includes a confidence threshold to avoid unreliable guesses.

H. Automated Quality Report Generation and User Interaction: The Zero-Error Workflow

The Human-in-the-Loop (HITL) interface empowers users to review detections and curate reports. Designed as the verification layer, it positions AI as an assistant, with the engineer selecting features for inclusion.

The interface supports intuitive interactions, such as cropping regions and labeling features.

Upon confirmation, the code generates a cleaned-up cropped image by erasing non-selected text and redrawing only chosen items with standardized labels, ensuring a clean, standardized visual output.

Minimizing disruption, the code generates Excel reports using a library to match existing formats, columns, and styles at the deployment site. Labeling conventions (e.g., X1, C1) align with company standards for familiarity and adoption.

A central orchestrator executes stages sequentially, applying exclusion zones and producing a comprehensive JSON file as the UI's data source.

Chapter IV. Implementation Details

A. Development Environment and Tools

This chapter documents the implementation of the software tool that constitutes the primary deliverable of this thesis. The tool was implemented using Python 3.10 as the primary programming language, chosen for its extensive ecosystem of libraries supporting data processing, computer vision, and web development. Python's interpreted nature facilitates rapid prototyping and debugging, making it ideal for iterative development in research-oriented projects. The code leverages a Read-Eval-Print Loop (REPL) environment for testing individual components, allowing incremental validation of functions like text extraction and symbol matching before full integration.

Key libraries included:

• PyMuPDF: A Python binding for the MuPDF library, used for parsing PDF files. It provides high-fidelity access to vector-based content, including text spans with bounding boxes and orientation data, enabling deterministic extraction without rasterization losses [16]. The code uses this library to iterate through document pages, extract blocks and lines, compute rotation angles from direction vectors, and normalize Unicode symbols for consistent matching.

• OpenCV-Python: An open-source computer vision library that offers tools for image processing and analysis [17]. It was employed for template matching, affine transformations, and non-maximum suppression to detect graphical symbols in rendered image regions. Implementation involves loading templates, applying multi-scale matching with correlation coefficients, and refining detections by suppressing overlapping boxes based on intersection-over-union thresholds.

• NumPy: A fundamental package for scientific computing in Python, providing support for large, multi-dimensional arrays and matrices, along with mathematical functions [18]. It was utilized for handling image data as arrays during vision processing stages, such as converting pixmap samples to arrays for grayscale conversion and coordinate transformations.

• Flask: A lightweight web framework for Python [19], used to create the backend server. It handles API endpoints for processing requests, enabling communication between the user interface and the core pipeline. The code defines routes for file uploads, cropping, labeling, and exporting, using JSON for data exchange and multiprocessing to run heavy computations in background threads without blocking the server.

• pywebview: A library for creating cross-platform desktop applications using web technologies. It embeds a web view to render the graphical interface, allowing the use of HTML, CSS, and JavaScript for a responsive UI without requiring a full browser [20]. The implementation spawns

a Flask thread and creates a resizable window to host the interface, ensuring platform-independent deployment.

• openpyxl: A Python library for reading and writing Excel files. It supports formatting, styling, and image embedding , making it suitable for generating quality reports that match existing templates. The code uses it to create worksheets, populate cells with selected data, apply borders and fills, merge cells for headers, and insert cleaned images for visual verification.

These tools were selected for their compatibility, open-source nature, and ability to run in a standalone desktop environment, ensuring the system could be deployed without external dependencies or internet access beyond the optional LLM API calls. Version control was managed implicitly through modular code structure, with error handling (e.g., try-except blocks for file operations) to gracefully manage exceptions like invalid PDF structures or missing dependencies.

B. Graphical User Interface (GUI) Workflow

The graphical user interface (GUI) serves as the primary interaction point for process engineers, designed to align seamlessly with their established workflow of curating inspection plans. The UI empowers engineers to review automated suggestions and select relevant features, shifting their focus from tedious manual data entry to strategic decision-making. This reduces the mental strain associated with cross-referencing drawings and spreadsheets, alleviating the cognitive fatigue from repetitive annotation tasks and allowing engineers to concentrate on higher-value activities like inspection planning and quality strategy.

The interface is implemented as a web-based desktop application, where the backend server manages state and processing while the frontend (built with HTML, CSS, and JavaScript) handles rendering and user inputs. Upon launching, the main UI presents a simple file upload section for paired PDF and IGES files, accompanied by a processing status indicator. The code validates uploads by checking file extensions and creates subfolders for organization, then initiates background processing via a multiprocessing process to avoid blocking the main thread.

Real-time feedback is provided through status icons (e.g., spinning loaders for "Processing..." and checkmarks for completion), updated via JavaScript polling of server endpoints.

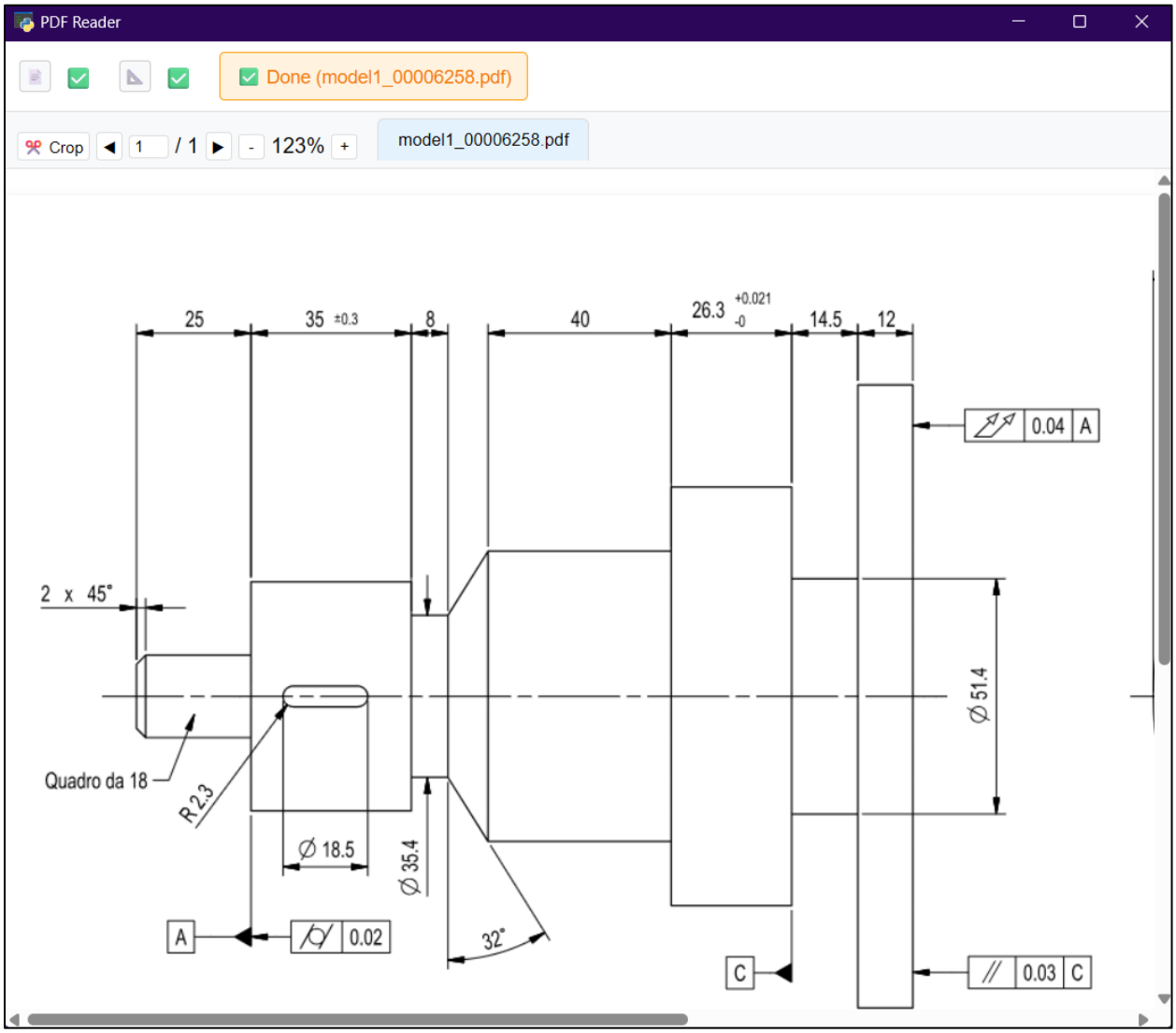

Figure IV-1 Initial Interface Screenshot

After processing, the interface displays a crop view of the drawing, overlaying detected annotations such as dimensions, tolerances, and GD&T symbols. The code loads the processed JSON data containing spans and matches, mapping them to interactive elements on the canvas. Engineers can zoom (using CSS transforms), pan (via drag events), and interact with these overlays to verify suggestions—clicking triggers dialogs for confirmation or editing. Coordinate transformations are handled in JavaScript to convert PDF coordinates to screen positions,

accounting for scaling, centering, and letterboxing to ensure accurate overlays on potentially resized images.

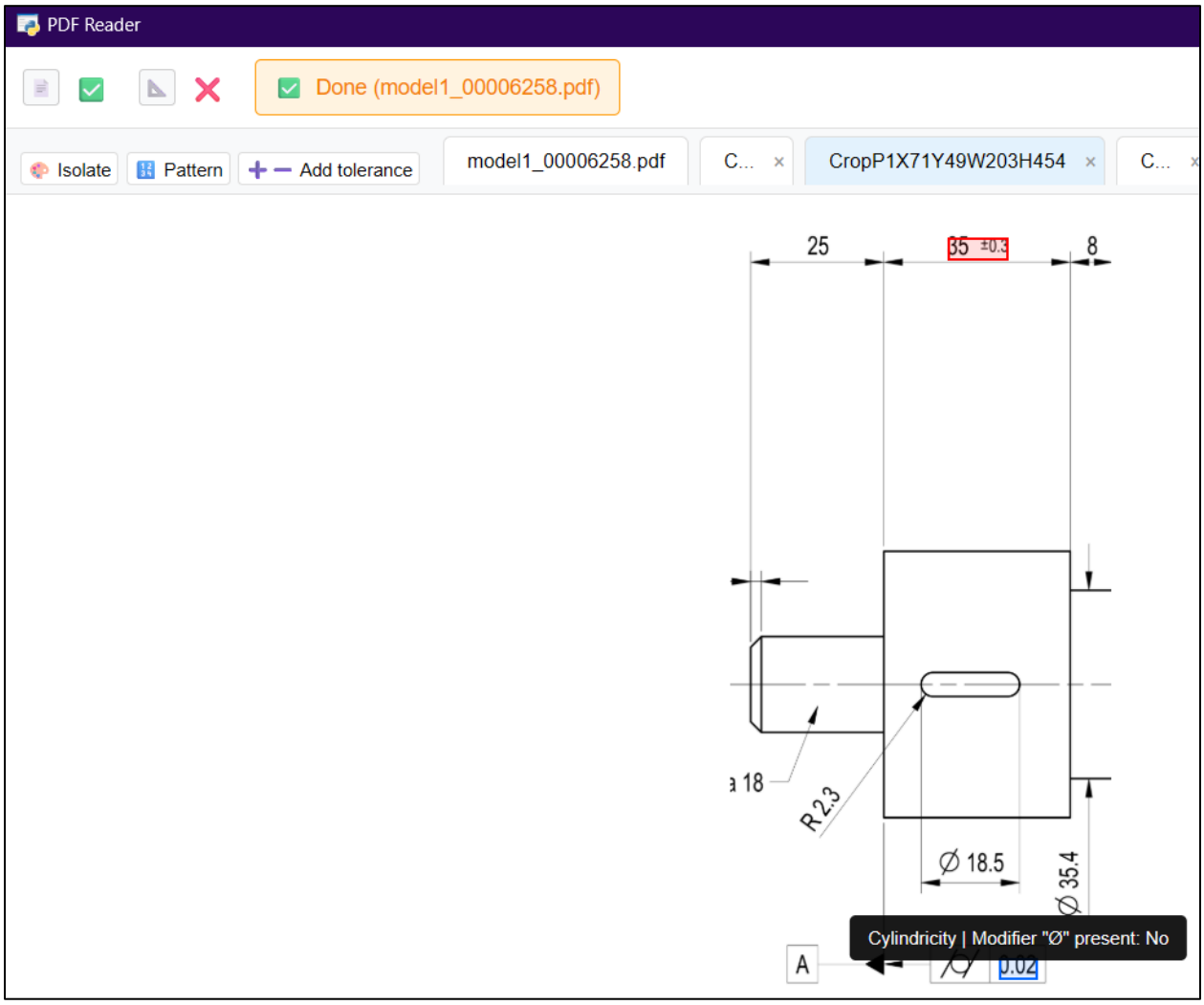

Figure IV-2 Interface Crop View with LLM suggested annotation

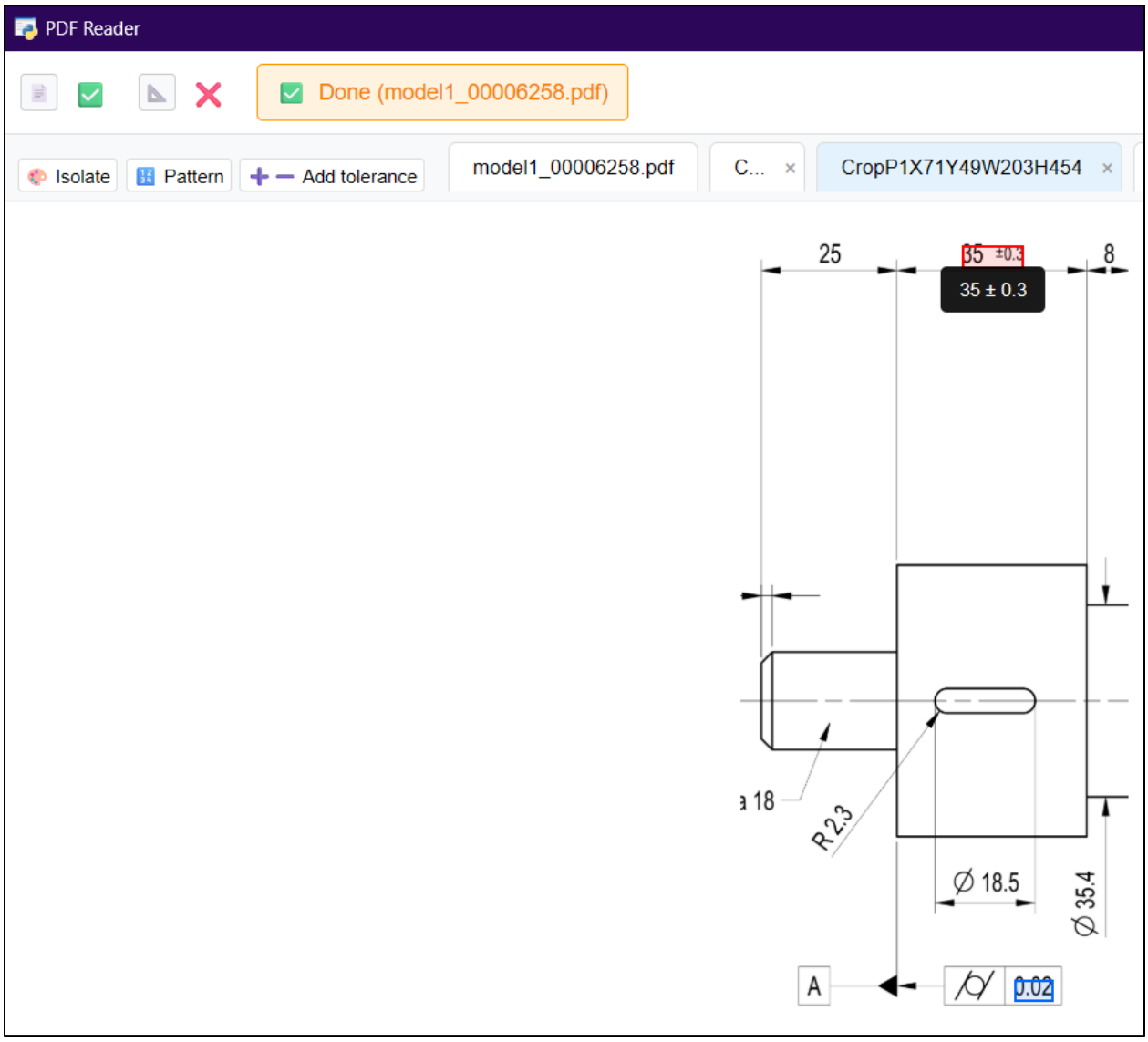

Figure IV-3 Interface Crop View with Dimension Annotation

The "Isolate" tool allows selection of individual features: the code registers click events on detections, opens a modal dialog for value confirmation or correction (e.g., adjusting tolerances), and assigns sequential labels (e.g., X1 for dimensions, C1 for conditions) using a generator function that cycles through letters and numbers. For repetitive elements like hole patterns, the "Pattern" tool enables grouping and batch labeling: users draw bounding boxes or select multiple

items, and the code groups them by proximity, applying batch labels and counting instances for report aggregation.

Manual additions are supported through an "Add Manual Tolerance" dialog, implemented as a form where users input base values, min/max tolerances, ISO grades, and optional diameter symbols. The code converts user-drawn bounding boxes from image to PDF coordinates (dividing by render scale), constructs new token groups, and appends them to the JSON data as manual entries with unique IDs.

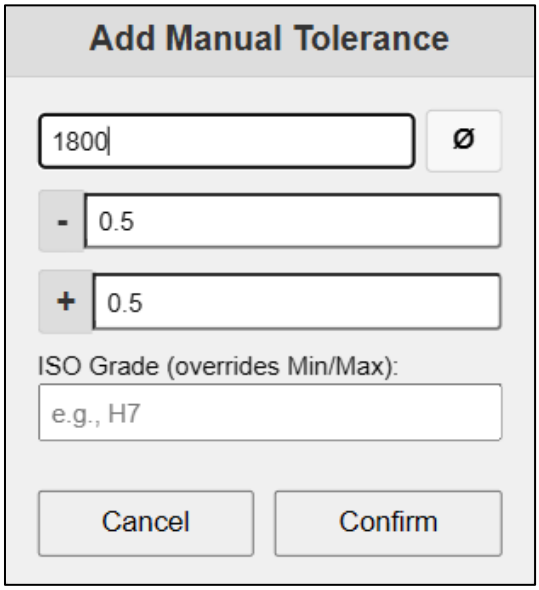

Figure IV-4 Adding Tolerance Interface

Upon finalization, the system generates a cleaned-up cropped image: the code loads the crop, erases non-selected text using drawing operations (e.g., filling white rectangles over unwanted areas), and redraws chosen elements with standardized labels via Pillow's ImageDraw, using fonts for clarity. The Excel report is created by initializing a workbook, populating sheets with dataframes from selected items (using pandas for structuring), applying styles (e.g., borders, alignments, fills), merging cells for headers, and embedding images. The code handles different data structures (isolated vs. patterns) by iterating through selections, styling cells programmatically, and saving with timestamps for uniqueness.

This workflow not only maintains structurally enforced output correctness through user selection but also relieves the mental burden of manual drawing annotation, such as repeatedly marking features or transcribing values under tight deadlines, fostering greater focus and job satisfaction. State management ensures buttons (e.g., "Isolate", "Export") are enabled/disabled based on processing status, with JavaScript updating the DOM dynamically.

C. Implementation of the Processing Pipeline

The core processing pipeline is orchestrated by a central coordinator that sequences the stages, handles data flow, and persists results in JSON for UI consumption. This implementation ensures modularity, with each stage building on the previous one's outputs, and includes error handling for file I/O and parsing exceptions.

1. IGES Parsing and Tokenization

The code begins by loading the IGES file lines, focusing on the parameter section marked by 'P'. It combines multi-line entries until a semicolon terminator, then parses entities starting with codes like "212," for general notes. Regular expressions extract quoted text, filtering out non-dimensional strings and identifying patterns such as diameter indicators ("1Hn"), GD&T codes (e.g., "1Hf" for flatness), and tolerances (e.g., "(\d+)H+([0-9.]+)" for positive values).

Tokenization converts raw dimensions into lists: for example, it splits "Ø1864H7 +0.1/-0.05" into ['Ø', '1864', 'H7', '+0.1', '-0.05'], handling unilateral/bilateral tolerances by checking signs and splitting on '/'. The code classifies tolerance types (e.g., "bilateral" for "+/-" pairs) and stores structured dictionaries with keys like 'dimension', 'tolerance', and 'raw_text'. Filtering excludes radii ("1HR") and empty tolerances, yielding a list of valid token groups sorted by complexity for prioritized matching.

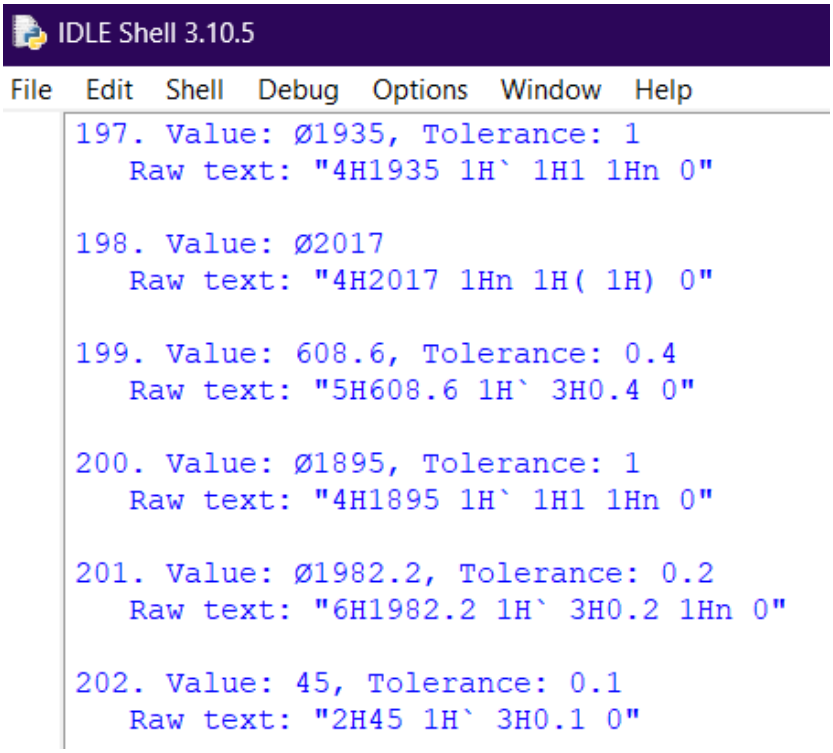

Figure IV-5 Debugging Step - Sample Token Output for Test IGES File

2. PDF Span Extraction and Normalization

PDF processing uses the library to open the document, iterate pages, and extract text dictionaries from blocks and lines. For each span, the code computes the rotation angle from the line's direction vector (using atan2 on cosine/sine components, normalized to 0-360 degrees), derives centers and dimensions, and normalizes symbols via a Unicode mapping (e.g., 'U+00D8' to 'Ø'). Spans are collected per page, with flags for later usage tracking to avoid double-matching.

Preprocessing organizes spans into a dictionary by page, adding 'used' flags initialized to False. This enables efficient filtering of available spans during matching, preventing overlaps.

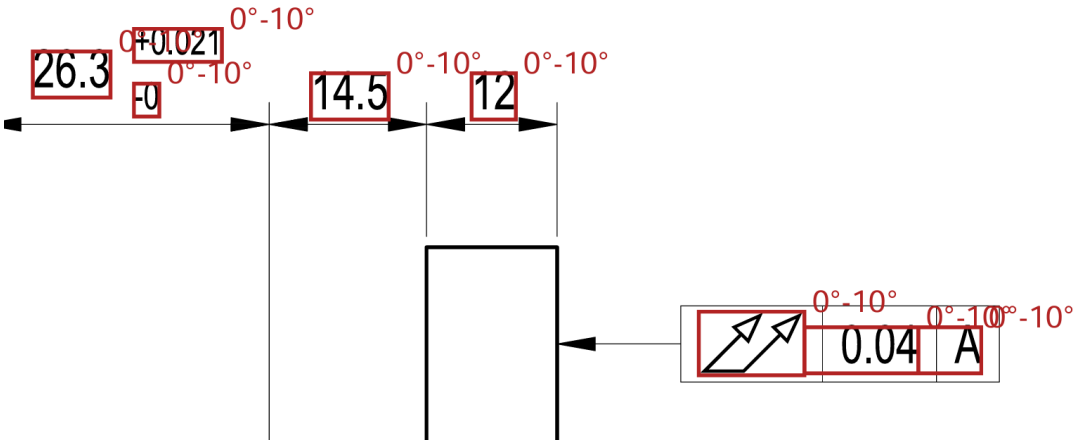

Figure IV-6 Extracted Spans for Sample Page with Angle Calculations

3. Multi-Modal Matching Algorithm

Matching iterates token groups from most to least complex. For each, the code generates combinations of available PDF spans matching token content (using counters for frequency), validates proximity (checking angle differences <1° and compact bounding boxes via max dimension thresholds), and scores candidates.

Scoring computes a composite metric: normalized area (smaller better) weighted at 50%, plus a reading order disorder score (Kendall tau distance on projections along the average angle's direction vector) at 50%. Projections use dot products of centers onto the unit vector (cos/sin of angle), sorting to compare against expected token order. The best match marks spans as 'used' and stores combined bounding boxes (excluding symbols like 'Ø' for tighter fits).

For unmatched groups lacking text-layer symbols, the code invokes vision detection in targeted regions, integrating results if matches are found (e.g., adding detected 'Ø' or '±' with confidence scores).

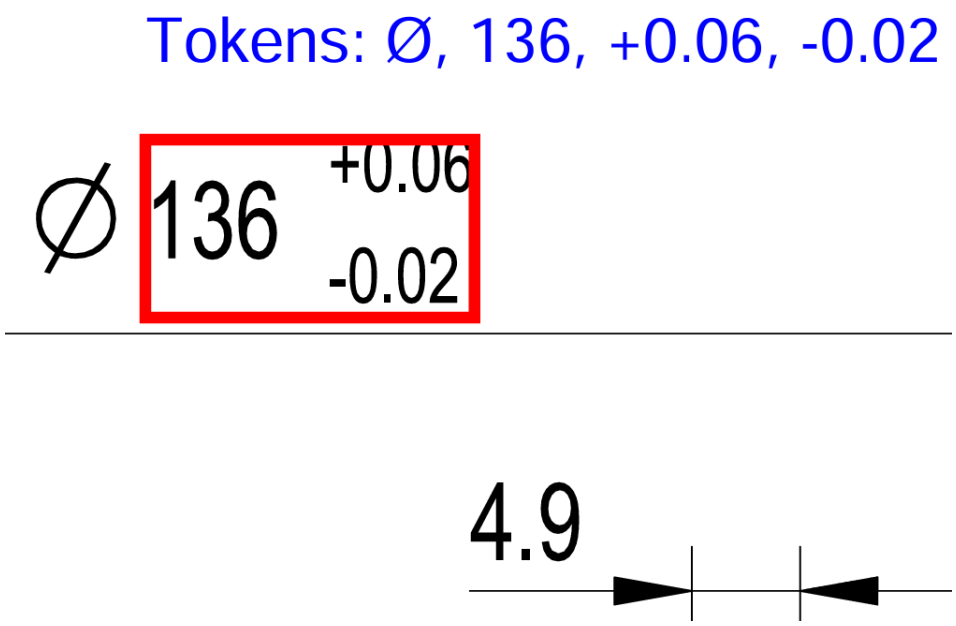

Figure IV-7 Debugging Step - Rendered Region and Detection Visualization

4. Computer Vision for Symbol Recognition

Symbol detection renders search regions (defined by calculated left positions using vector math: local coordinates rotated by text angle, then axis-aligned bounding boxes) as high-resolution pixmaps, converted to grayscale arrays. Images are rotated to horizontal via affine matrices (getRotationMatrix2D with center adjustments), preserving boundaries.

Template matching loads predefined images (e.g., for 'Ø' and '±'), scales them (0.5-1.5x), and applies TM_CCOEFF_NORMED correlation. Thresholds (0.75+) filter locations, with non-max suppression grouping by IoU (>0.3) and selecting highest scores. Downsampling (e.g., 0.75 factor) trades minor precision for speed, caching rendered pages to avoid redundant computations.

Detections return global PDF coordinates by transforming back via inverse affine matrices, with debug images optionally saved showing boxes and scores.

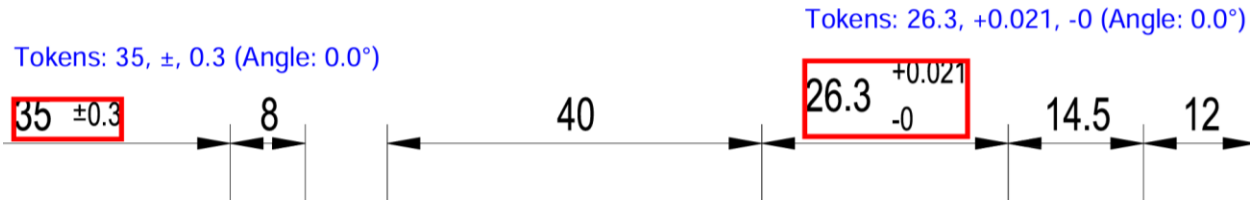

Figure IV-8 Example of a Detection Visualization

5. Unmatched Values and GD&T Frame Extraction

Post-matching, the code filters remaining spans for standalone numerical conditions (e.g., '0.1' via regex for decimals <1 or integers 1-5, plus patterns like "/100"). Dynamic multipliers (10x for single chars, scaling to 1.5x for longer) define search widths, with heights 1.5x text height.

Search areas use rotation-aware calculations: local rectangle corners rotated and translated, yielding polygons for masking. Regions are masked, cropped tightly (finding non-white bounds), and saved as horizontal PNGs with metadata filenames (e.g., "pageN_XposYpos_WwidthHheight_text_angle.png"). Exclusion zones (expanded matched bboxes) prevent re-processing.

D. AI Integration for GD&T Classification

The integration of AI for classifying GD&T symbols from extracted image snippets is implemented through a cloud-based LLM API, specifically using a vision-language model to analyze visual content combined with textual prompts. The code formats the prompt by injecting the condition value (e.g., the numerical tolerance like "0.1") into the template, which includes task definitions, a symbol dictionary, disambiguation rules, chain-of-thought instructions, and structured output requirements. This is done via string formatting in the PROMPT_TEMPLATE constant, ensuring the prompt is self-contained and guides the model to output in a parsable format like "[Symbol Name] | Modifier 'Ø' present: [Yes/No]" or "Unclear" for low-confidence cases.

API calls are managed in a dedicated function that increments a global counter (api_call_count) for tracking usage, constructs the multi-part request with the image file and text prompt, and handles HTTP POST to the endpoint. Response handling parses the JSON output, extracting the symbol name and diameter presence while validating against expected formats; invalid responses trigger fallbacks to "Unclear." The code updates the JSON data structure by appending classification results to unmatched values, including confidence thresholds to filter unreliable outputs (e.g., if the model explicitly returns "Unclear")

Error handling wraps the API interaction in try-except blocks to catch network issues or rate limits, logging failures and proceeding with manual user overrides in the UI if needed. This implementation ensures robustness, with retries for transient errors and batching for multiple snippets to optimize calls. Debugging placeholders in the code allow printing prompt-response pairs for verification during development.

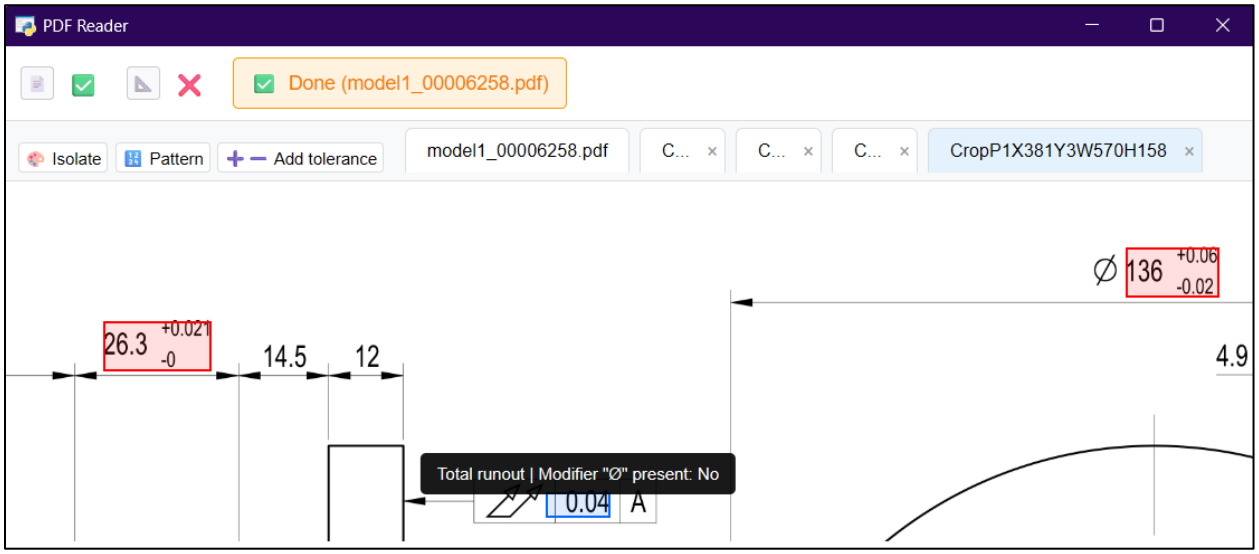

Figure IV-9 Example of a Returned Value from the Gemini LLM API

E. Automated Quality Report Generation

Report generation is implemented in the backend endpoints for exporting single crops or all data, using openpyxl to create or modify Excel workbooks that mimic enterprise templates. The code initializes a new workbook or loads an existing one, creates sheets for dimensions and conditions, and populates them via pandas dataframes converted to rows (using dataframe_to_rows for efficient insertion). For each selected item, it structures data into columns like "Label" (generated sequentially via _generate_sequential_label or _generate_condition_label functions, which cycle through letters like X1-9, Y1-9), "Nominal" (base values with optional 'Ø'), "Tolerance" (formatted as "+max/-min" or unilateral), "GD&T Symbol" (from classifications), and "Notes" (raw text or descriptions).

Styling is applied programmatically: headers get bold fonts, gray pattern fills, and centered alignments; borders use thin sides; cells with tolerances merge if patterned and images are embedded via OpenpyxlImage, anchored to specific cells. The code handles patterns by replicating rows with incremented labels and aggregating counts, ensuring merged cells span appropriately. Filenames include timestamps to avoid overwrites, and platform-specific commands (os.startfile on Windows, subprocess with "open" on macOS, "xdg-open" on Linux) open the file post-save.

This implementation relieves cognitive load by automating formatting that engineers previously did manually, with error checks for missing data (e.g., skipping unclassified symbols) and logging for export failures.

F. System Optimization and Error Handling

System optimizations are embedded throughout the code to ensure responsive performance in a desktop environment. Asynchronous processing uses multiprocessing.Process to run the pipeline in a separate thread, preventing UI freezes during heavy computations like matching or rendering; the main thread polls for completion via shared flags or JSON file watches. Caching renders PDF pages at reduced resolution (downsample_factor=0.75) in a dictionary, reusing grayscale arrays for repeated accesses in symbol detection or cropping. Template loading for matching pre-scales images (0.5-1.5x) and limits detections per template to 2, reducing computational overhead.

Efficiency heuristics include sorting token groups by length (descending) to match complex items first, dynamic thresholds (e.g., proximity multipliers scaling with text length), and exclusion zones expanded by 10% to minimize redundant checks. Batch processing for LLM calls groups snippets, though limited by per-user setups in deployment.

Error handling is comprehensive: file operations use with-blocks and os.path checks; parsing wraps in try-except for ValueErrors (e.g., invalid angles) or IndexErrors (e.g., malformed IGES); API responses validate formats with regex, defaulting to "Unclear"; UI endpoints return JSON errors (e.g., 404 for missing files, 500 for exceptions) with messages for user alerts. Logging via logging.basicConfig captures debug/info/error levels, aiding post-mortem analysis. Deployment challenges, like per-PC setups, are mitigated by standalone executables, but shared database integration is noted for future scalability.

Chapter V. Results and Discussion

A. Case Study: Deployment at Ansaldo Energia

The system's effectiveness was validated through an industrial case study. The deployment involved integrating the tool into existing quality assurance workflows, where engineers routinely generate inspection reports from mechanical drawings as part of standard industrial practice.

The evaluation used a diverse dataset of 10 technical drawings, selected to represent varied complexity levels. These included drawings with different tolerance grades, symbol densities, and geometric complexity levels, representative of typical precision manufacturing documentation. The system processed paired PDF and IGES files for each, extracting dimensions, tolerances, and symbols, followed by user selection and report generation.

Results confirmed the system's structurally enforced output correctness: because engineers select only the features they intend to inspect, the architecture makes it impossible for unverified data to enter the final report, addressing the transcription errors characteristic of manual processes. User adoption was immediate: the intuitive interface and alignment with familiar spreadsheet templates and labeling conventions meant that no formal training was required beyond the visual guide provided with the software. Engineers confirmed seamless integration with their existing document workflows, with no modifications required to downstream processes, validating the non-invasive design philosophy.

B. Performance Analysis

Performance was assessed through quantitative metrics for extraction stages and comparative benchmarks against manual methods, drawing from user reviews and time-motion observations during the case study.

1. Quantitative Results

The automated extraction pipeline showed strong initial suggestion accuracy. The deterministic matching of IGES tokens to PDF spans achieved approximately 98% accuracy across the dataset, correctly associating most dimensions and tolerances based on proximity and order heuristics. Symbol detection via targeted computer vision correctly identified about 80% of GD&T elements, with failures primarily in low-quality raster inputs or highly overlapped regions. These rates significantly reduced the need for manual corrections, streamlining the selection process. Regarding final report accuracy: the architecture enforces output correctness by construction. Users define the report scope through mandatory selection, and any AI suggestion not explicitly confirmed is excluded. The final report therefore reflects deliberate human choices, not unverified automated outputs — a structural advancement over tools lacking a validation layer.

2. Comparative Analysis

A time-motion analysis informed by user observations and feedback across the 10 case study drawings consistently identified drastic reductions in report generation time. The primary driver was the replacement of manual symbol-by-symbol searching and transcription with a guided selection workflow: automated suggestions reduced the effort of locating and entering each data point, while the structured GUI eliminated cross-referencing between separate applications. Variation across drawings depended on symbol density and complexity, but the directional finding was consistent.

From a cost perspective, while formal economic quantification was outside the scope of this study, the in-house architecture avoids the recurring usage fees associated with commercial APIs such as Werk24, offering a more sustainable cost model for high-volume industrial deployment. The open-architecture design also eliminates vendor lock-in and enables domain-specific customization unavailable in proprietary tools.

On the dimension of cognitive workload, engineers described the pre-existing manual process as mentally demanding: it required sustained vigilance during repetitive transcription, constant cross-referencing between PDF viewers and spreadsheet templates, and careful verification at each step to avoid cascading errors. The tool re-allocated this low-level effort to the system, freeing engineers to focus on higher-value decisions — interpreting design intent, prioritising inspection scope, and assessing tolerance criticality. Direct feedback from the quality team underscored this cognitive relief: automated overlays and one-click selections were cited as the primary source of time and effort savings, allowing quicker iterations and fewer revision cycles compared to the manual workflow.

C. Quality Report Generation Evaluation

The generated quality reports were evaluated for accuracy and usability, comparing them to manually created ground-truth versions from the case study drawings.

Report Accuracy: All system-generated reports matched ground-truth content exactly for selected features, with structured Excel outputs including columns for nominal values, tolerances, GD&T classifications, and embedded cleaned images. Discrepancies were absent due to the selection workflow, confirming high fidelity.

User Experience and Workflow Integration Evaluation: Qualitative responses confirmed immediate usability and seamless fit with existing downstream processes. The high adoption rate stemmed from the system's non-invasive design: no changes to existing workflows were required, and engineers consistently highlighted relief from annotation strain as a primary benefit.

D. Discussion

1. Interpretation of Findings

The quantitative and qualitative results presented above confirm the viability of the proposed approach. The key finding is that by restructuring the human operator's role — from error-prone transcriber to deliberate selector — the system achieves both high accuracy and high usability simultaneously. These results are discussed in the context of Industry 4.0 implications and current limitations in the subsections below.

2. Implications for Industry 4.0

This system advances manufacturing's digital transformation by bridging the "data dichotomy," creating a structured digital thread from drawings to inspections. It supports Model-Based Enterprise goals by converting static documents into interoperable data, enabling real-time QA in smart factories. Broader adoption could disrupt technical document processing, promoting scalable, human-centric automation across sectors like aerospace and automotive.

3. Limitations

Despite successes, limitations were noted. The reliance on a cloud-based LLM API for GD&T classification introduces cost and scalability concerns, with performance varying by PDF quality (e.g., raster vs. vector). A key deployment challenge was the inability to recreate the LLM API endpoint for every user, as the system was not implemented with a shared database infrastructure, requiring individual configurations. Setup on each desk PC proved tedious, involving environment installations and dependency management, hindering enterprise-wide scaling without further refinements like containerization. These factors underscore areas for future optimization to enhance accessibility and robustness.

Chapter VI. Conclusions and Future Work

A. Summary of Contributions

This thesis has successfully met its objectives by designing, implementing, and validating a functional software tool — a hybrid intelligence desktop application — for automating quality reporting in precision manufacturing environments. The tool is the central and concrete output of this work; the contributions described below are realized through it, not merely proposed in theory. The primary objective—to detect and extract dimensions, tolerances, tolerance grades, associated symbols (e.g., Ø, ±), and unidentified GD&T feature control frames from paired IGES and PDF drawings—was achieved through a multi-modal pipeline that correlates structured IGES data with PDF vector spans, augmented by targeted computer vision and LLM-based classification. The secondary objective—integrating these detections into an automated report generation system with human-in-the-loop verification—was realized via an intuitive desktop interface that ensures seamless compatibility with Industry 4.0 frameworks, as demonstrated in the Ansaldo Energia case study. The tertiary objective—showcasing a cost-effective "glass-box" architecture as an alternative to opaque commercial solutions—was validated by the system's transparent, deterministic processes that avoid recurring fees while delivering superior functionality.

The primary contribution is a novel, implemented software tool embodying an architecture that redefines the human operator's role in AI-assisted workflows: from a fallible validator of AI suggestions to a deliberate selector of report scope, structurally preventing unverified data from entering the output. Additional contributions include a working, open-architecture alternative to expensive commercial "black-box" solutions, and a demonstrated non-invasive integration methodology validated in a live industrial setting.

Innovations include the multi-modal matching algorithm, which heuristically scores text groupings based on spatial proximity, orientation, and semantic order for precise correlations; the targeted computer vision module for symbol detection in defined regions; and the LLM-based GD&T classification workflow, enhanced by engineered prompting for robust, machine-parsable outputs. These elements collectively form a complete, end-to-end solution that enhances efficiency and accuracy in quality assurance.

B. Conclusions and Commercial and Industrial Impact

In conclusion, this project addresses a critical inefficiency in manufacturing quality assurance by automating the extraction and reporting of dimensional and GD&T data from mechanical drawings. Key takeaways include the proven viability of a hybrid intelligence approach that combines deterministic parsing, heuristic matching, targeted vision, and mandatory human selection to verified outputs, significant and consistent time savings per drawing, as reported by

users, and substantial relief from cognitive workload associated with manual annotation. The industrial case study confirms the system's practical viability, with immediate user adoption and seamless workflow integration demonstrating its readiness for deployment in precision manufacturing environments.

Commercially, the solution has market disruption potential in the technical document processing sector, offering a scalable, open-architecture alternative to proprietary tools. Its high applicability extends across manufacturing sectors, from power generation to aerospace, enabling tangible transformations in quality workflows by bridging the "data dichotomy" and fostering a structured digital thread. Industrially, it advances Industry 4.0 goals by promoting data-driven, human-centric automation that reduces lead times, minimizes errors, and enhances engineer productivity through cognitive relief.

C. Future Research Directions and Future Work

The current implementation, while validated in a live industrial environment, represents a foundational version of a more expansive system. Several directions for future development were identified during the case study and are outlined below. These range from infrastructure improvements that would address the deployment limitations noted in Section V.D, to substantive extensions of the system's analytical capabilities.

1. Cloud-Based SaaS Deployment

Transition to a multi-tenant SaaS platform via cloud migration, using a centralized database for shared access, secure authentication, and automatic updates to enable distributed teams and eliminate per-user setups.

2. Deep Enterprise Integration

Enhance the backend with API endpoints for automated triggering from PLM systems, background processing, and data pushes to MES for real-time inspection scheduling and resource allocation, advancing toward "lights-out" automation.

3. Intelligent Selection Learning

Develop a lightweight ML model to predict annotation relevance from historical selections, ranking suggestions in the UI to accelerate verification while preserving human validation and structurally enforced output correctness.

4. Advanced Symbol Detection Models

Train compact object detection models on user-validated data generated by the system, replacing template matching to boost accuracy for distorted or overlapped symbols, maintaining efficiency for real-time use.

5. Generalized Model Development: Cross-Industry Applicability

Leverage the annotated dataset for training generalized models applicable beyond power generation, fostering industry-standard tools for dimensional inspection with enriched contextual understanding.

References

[1] "Almada-Lobo, Francisco. (2016). The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES). Journal of Innovation Management.".

[2] "Nunes, Tamires & Zanini, Roselaine & Ferreira Porto Rosa, Ariane & Vergara, Lizandra. (2022). Impacts and challenges of industry 4.0 in manufacturing: a systematic literature review. The Journal of Engineering and Exact Sciences. 8. 16294-01e. 10.18540/jc," 2022.

[3] "Lipman, Robert & Filliben, James. (2020). Testing Implementations of Geometric Dimensioning and Tolerancing in CAD Software. Computer-Aided Design and Applications. 17. 1241-1265. 10.14733/cadaps.2020.1241-1265.," 2020.

[4] "Sauer, C.R., Burggräf, P. Hybrid intelligence – systematic approach and framework to determine the level of Human-AI collaboration for production management use cases. Prod. Eng. Res. Devel. 19, 525–541 (2025). https://doi.org/10.1007/s11740-024-01326-7," 2025.

[5] "D. Dellermann, N. Lipusch, P. Ebel, and J. M. Leimeister, "Design principles for hybrid intelligence decision support systems," arXiv preprint arXiv:2105.03354, 2021.," 2021.

[6] "M. Humienny, "Overview of Principles and Rules of Geometrical Product Specifications According to the Current ISO Standards," Advances in Science and Technology Research Journal, vol. 17, no. 5, pp. 1-12, 2023. doi: 10.12913/22998624/169.," 2023.

[7] "ISO 8015:2011, "Geometrical product specifications (GPS) — Fundamentals — Concepts, principles and rules," International Organization for Standardization, 2011.," 2011.

[8] "Vector vs Raster files," [Online]. Available: https://reformcreative.co.uk/vector-vs-raster-files. [Accessed 01 09 2025].

[9] "Mustafa, Faiz & Al-Ashaab, Ahmed & Al-Amili, Hussein. (2017). A Comparative Study of Product Data Exchange among CAD Systems.," 2017.

[10] "Blog / GD&T symbols," [Online]. Available: https://www.school-mechademic.com/blog/gd-t-symbols. [Accessed 01 09 2025].

[11] "Moreno-García, Carlos & Elyan, Eyad & Jayne, Chrisina. (2019). New trends on digitisation of complex engineering drawings. Neural Computing and Applications. 31. 10.1007/s00521-018-3583-1.," 2019.

[12] "Redmon, Joseph & Farhadi, Ali. (2018). YOLOv3: An Incremental Improvement. 10.48550/arXiv.1804.02767.," 2018.

[13] "Xie, Jiarui & Sun, Lijun & Zhao, Yaoyao. (2024). On the Data Quality and Imbalance in Machine Learning-based Design and Manufacturing—A Systematic Review. Engineering. 45. 10.1016/j.eng.2024.04.024.," 2024.

[14] "Opto Engineering, Template Matching," [Online]. Available: https://www.opto-e.com/it/basics/template-matching. [Accessed 01 09 2025].

[15] "Sahoo, Pranab & Singh, Ayush & Saha, Sriparna & Jain, Vinija & Mondal, Samrat & Chadha, Aman. (2024). A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. 10.13140/RG.2.2.13032.65286.," 2024.

[16] "Extracting and Creating Vector Graphics in a PDF Using PyMuPDF,," Artifex Software, [Online]. Available: https://artifex.com/products#pymupdf. [Accessed 09 01 2025].

[17] "Bradski, Gary. (2000). The Opencv Library.," 2000.

[18] "van der Walt, Stéfan & Colbert, S. & Varoquaux, Gael. (2011). The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science & Engineering. 13. 22 - 30. 10.1109/MCSE.2011.37.," 2011.

[19] "Welcome to Flask — Flask Documentation (3.1.x)," [Online]. Available: https://flask.palletsprojects.com/en/stable/. [Accessed 01 09 2025].

[20] "pywebview Documentation," [Online]. Available: https://pywebview.flowrl.com/. [Accessed 01 09 2025].

[21] "Working with styles — openpyxl 3.1.3 documentation," [Online]. Available: https://openpyxl.readthedocs.io/en/stable/styles.html. [Accessed 01 09 2025].

[22] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," IEEE Access, vol. 6, pp. 52138–52160, 2018. DOI: 10.1109/ACCESS.2018.2870052

[23] M. Riesener, C. Dölle, and C. Lichtenberg, "Framework for Quality Assurance of Machine Learning Applications in Manufacturing," Procedia CIRP, vol. 93, pp. 1010–1016, 2020. DOI: 10.1016/j.procir.2020.04.004

[24] R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice. Wiley, 2009. ISBN: 978-0-470-51706-2

[25] ISO 1101:2017 — Geometrical product specifications (GPS): Geometrical tolerancing — Tolerances of form, orientation, location and run-out. International Organization for Standardization, Geneva, Switzerland, 2017. Available: https://www.iso.org/standard/66777.html